A Study on Internal Operation Mechanism of Mobile Software Ecosystem Based on Computational Experiments

***Abstract*—In addition to publishing and downloading mobile apps, Mobile App Store (MAS) has become the most important ecosystem on mobile smart devices, i.e., Mobile Software Ecosystem (MSECO). However, most of the existing work focus on the analysis of a single entity in MSECO, and rarely analyze the interaction between the entities (such as users, developers, etc.) in MSECO as well as the comprehensive effect of each entity to the entire ecosystem. In this paper, we propose a method based on computational experiments to simulate and analyze the complex and dynamic interaction of various entities in MSECO. Firstly, a requirement-driven MSECO model named R-MSECO is established to break down the entire MSECO, which includes user-app interaction, developers-requirements interaction and macro control of mobile platform three sub-models. Secondly, the idea of "computational experiments" is exploited to simulate the dynamic interaction process between various entities in MSECO. Finally, the simulation results and comparative analysis of real cases to verify the validity and feasibility of the model. The experimental results show that our method is helpful for mobile users to better understanding of the current state of MSECO as well as can provide a reference for developers and platform managers to make development and operation decisions, which is of great significance for the continued healthy development of MSECO.**

***Keywords*-Mobile Software Ecosystem, Requirement-Driven, Computational experiment, Complex System**

I. Introduction

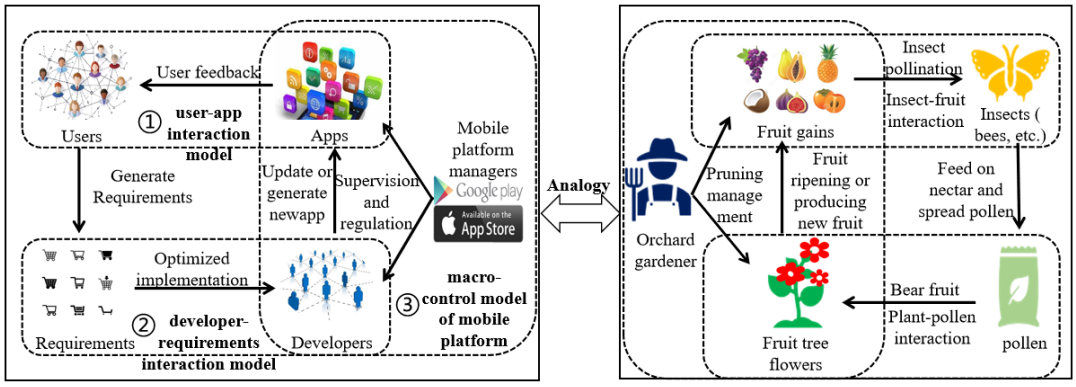
MSECO is a set of a collaborative systems, users and developers that creates complex relationships driven by competition and cooperation within niches-similar to biological ecosystems [1]. MAS (such as Apple App Store) have achieved great success as a commercial innovation concept of mobile software, which has greatly changed the way of users to buy, use and comment software [2]. currently, MAS is the most typical MSECO display form. Therefore, an in-depth study of the MSECO internal operation mechanism is of great significance for the healthy development of the mobile app market.

Awdren Fonto et al. [1] described MSECO as a set of collaborative systems which include platforms, users, developers, communities, applications, mobile apps, and evangelist entities. This is the first time when researchers have officially defined the MSECO. The current research on MSECO is still in its preliminary stage. Steglich C et al. [3] systematically analyzed the current literature on MSECO and found that almost 50% of the publications are from 2015, and the research on MSECO has been in a growth period and is receiving increasing attentions from both industry and research community. However, most of these studies have stayed at the level of defining the concept of MSECO [3], or focused on a single entity in MSECO, such as study the developers [4][5][6], mobile platform repository management [7], etc. They either just summarize the empirical analysis of the user and platform data [8], or discuss the working mode of the platform [9][10]. To our best knowledge, there is rare work on modelling and analyzing MSECO from a macro perspective as a whole.

In fact, there are always dynamic interactions between various entities in MSECO. For example, users continuously search, download and evaluate apps on mobile platforms. Developers continuously update and maintain their own apps to obtain their own benefits. And the platform managers continue to open and maintain their infrastructure to attract external developers to enter their platform to publish various apps, which try to meet the users' requirements [7]. This is an extremely complex and dynamic ecosystem, and thus studying its inner working mechanism remains a challenging task.

In this paper, we model and analyze the dynamic interaction of various entities in a real mobile software ecosystem. We propose a requirement-driven MSECO model named R-MSECO, which includes user-app interaction, developer-requirement interaction, and macro control of mobile platform as the three sub-models. Then, we leverage the computational experiment method to establishing a computational experimental system for the R-MSECO from a global perspective, and then gradually analyze the dynamic interactions between various entities in MSECO. The experimental results show that R-MSECO can well represent the complex internal dynamic interaction mechanism with multiple entities in MSECO. This work can be used as a test bed to explore new ideas and new business model for the healthy and sustainable development of MSECO. The contributions of this paper are highlighted as follows:

- A requirement-driven MSECO model named R-MSECO is proposed which take the users' requirements as the internal core driving force to driving the dynamic operation of the entire MSECO.
- We first introduce the approach of "computational experiment" to simulating the dynamic and complex interaction process of multiple entities in MSECO.
- We analyze and verify the effectiveness and feasibility



(a) R-MSECO model (b) Orchard Ecosystem

Figure 1. R-MSECO analogy with orchard ecosystem in nature

of R-MSECO with real world data of typical MSECO.

The rest of the paper is structured as follows: Section 2 details the overall architecture of R-MSECO. Section 3 describes the key entities of R-MSECO and their corresponding attributes and behaviors and Section 4 shows and discusses the results of simulation experiments. Section 5 analyzes the validity and feasibility of the model through real case verification. Section 6 discusses the related work and Section 7 summarizes and outlines the future work.

II. R-MSECO: A Requirement-Driven Model for MSECO

In the natural world, we can classify the species in the ecosystem into several roles, each of which has a different division of labor in the ecosystem, thereby forming a relatively closed and self-regulating circle. There are many similar characteristics with MSECO. Inspired by this, we establish a requirement-driven MSECO model R-MSECO as shown in Figure 1(a).

R-MSECO can be analogized to the typical orchard ecosystem in nature(as show in Figure 1 (b)), where the users correspond to insects that eat nectar, and the developers correspond to the plants (flowers), the requirements correspond to pollen, the apps correspond to fruits, and the platform managers correspond to orchard gardeners. The process of user-app interaction (such as user group feedback) to generate requirement corresponds to insects eating the flowers in fruit tree and at the same time the pollen was spread. The process of developers interacting with requirements to update or generate new apps is equivalent to plants getting pollen to grow new fruits or the fruits gradually grow up and mature, and the process of supervision and regulation by mobile platform managers is equivalent to the process of orchard gardeners managing fruits and fruit trees. R-MSECO is a complex and dynamic interactive system. In order to better portray and understand its internal operating mechanism, we need to gradually simplify a complex problem, i.e., to subdivide the entire R-MSECO into smaller components. In this way, the large and complex problems can be transformed into a series of small and simple problems. The definition of MSECO can be referred to literature [1], and we can describe the entity elements involved in R-MSECO as follows:

Users: Search and download apps to meet their requirements through mobile platforms, and meanwhile the users can provide feedback to the developers about the app or promote the financial growth of the app by purchasing.

Developer: Refers to the group that is registered on the mobile platform to pursue the return on value, they develop the app and publishes it to the mobile platform by satisfying the users' requirements.

Mobile App: An online product developed by the developer and approved by mobile platform managers and deployed on a mobile platform with the functions that meeting the users' requirements.

Mobile platform (hereinafter referred to as platform): An online store where users can search and acquire software solutions (apps). The difference between a platform and other entities (e.g., users and developers) are that the platform is unique, while other entities can be multiple. In addition, the goal pursued by the platform is the positive and healthy development of MSECO.

Requirement: It mainly refers to user's value claims. The requirement is an abstract content and it is also an indispensable element of the entire MSECO, it is the core driving force that drives the internal operation of MSECO.

As shown in Figure 1(a), we divide R-MSECO into three sub-models according to the different entity interaction activities, i.e., the user-app interaction model (generating requirements), and the developer-requirements interaction model (competition and collaboration are exist between developers) and the macro-control of the model platform. These three sub-models are not independent but closely related.

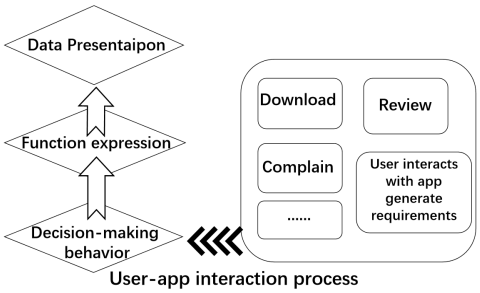


Figure 2. Three-layer model of user-app interaction mechanism

A. User-App Interaction Model

The user-app interaction model is a core sub-model in the R-MSECO since it is one of the main processes to generating user requirements. In a complete R-MSECO, there are multiple users and apps. In addition, during the interaction between a specific app and its users, a large amount of behavioral data is generated, and these data are statistics by the mobile platform. In fact, the user-app interaction process is actually an activity performed by the users in order to meet their own requirements. These users has the power to make decisions, and the data was generated by the users' decision. This can essentially explain the inherent driving force in MSECO. In order to express this process more detail, as shown in Figure 2, we further layered the user-app interaction model into the decision-making behavior layer which the user decides his behavior, function expression layer which explains the intermediate process of decision-making turn into platform data, and the data presentation layer which represents the data displayed on the platform and converted into user's requirements, where the decision-making layer is the cause and the presentation layer is the effect.

B. Developer-Requirement Interaction Model

In the process of developers-requirement interaction, the developers will continue to check the users' requirements based on users' feedback in the app market. This is a requirement-driven app update and maintenance based on user group feedback. Although there are other motivations for developer to updating apps, but they are essentially to meet the users' requirement. And it is worth noting that the presentation layer divides the data under the platform and the data on the platform. The data under the platform are the developer's attributes which can be calculated or be set to, and the data on the platform represent the data updates on the platform, such as update records and software descriptions that after the developers have implemented the requirements.

C. Macro Interaction Model of Mobile Platform

In the macro control of mobile platform model, the platform manager mainly plays two roles. One is the platform supervisor. The platform needs to process users feedbacks on the app in a timely manner. These feedbacks may express the user's requirements. Another role of the platform is the regulator of MSECO. The platform needs to formulate various strategies to promote the healthy development of MSECO, such as formulating developer development specifications, providing development resources, testing environments or recommending high-quality apps.

Table I
ATTRIBUTES OF USER AGENT

Attribute	Description
Expect	User expectations UEA=$\{UE^1, UE^2, .., UE^m\}$ which represents the expectations of different users for the app, ESQ, EPD
Group	According to user preference, different users can belong to different group
DownloadRecord	App download records of user

Table II
BEHAVIORS OF USER AGENT

Behavior	Description
Download	if app (downloads, evaluation, awareness) >User's expectation {downloads, evaluation, awareness}, then download, else not download
Rate	Scoring function and ESQ, there is a positive correla tion between rate and user's ESQ
Complain	If the asq <limit value (the asq is lower than a certa in value), and the probability R(c) which expressed as a proportion of user complaints is too large, then com plaint
Generate Requirement	If ESQ -ServiceQuality>0 then Generate Requirement

III. ATTRIBUTES AND BEHAVIORS OF MODEL ENTITY

A. User Agent

User Agent can be described as a set of attributes and behaviors. Table I shows the attributes of the User Agent, where the user expected value UEA= {App service quality ESQ, Platform data EPD}, in which ESQ= {Throughput, ResponseTime, Integreity, UI, usablility, Security, Maintainability, etc.}, EPD={Download, Rate, CognitionDegree, etc.}.

The user's behavior is mainly the user's decision making behavior which as show in Table II. Among them, the service quality of app (asq) is defined as ServiceQuality, Rate = =W* (Expect(asq)-ServiceQuality), W is the normalization parameter, Expect(asq) is the quality of service expected by users.

B. Requirement Agent

Requirement do not need to make feedback behaviors on external event stimuli, but requirements need to have their own attributes, which will change over time. Table III shows

Table III
ATTRIBUTES OF REQUIREMENT AGENT

Behavior	Description
RequirementExpect	In R-MSECO, the requirement expectation has a linear correlation with the app service quality th at users expect, i.e., weight = 1
RequirementState	1= satisfied, 0=unsatisfied

Table IV
ATTRIBUTES OF DEVELOPER AGENT

Attribute	Description
Energy	The initial energy of the developer is randomly generated, which is subsequently affected by the learning behavior. The sum of the app service quality = the developer's current energy*weight, App.ServiceQuality=Energy*weight
Cost	Cost = Function(Action,Scale), The cost required by the developer's behavior cost = function F (scale, behavior) The larger the size of the developer, the greater cost for his behavior
Scale	The developer scale is determined by the total capital, the more capital, the larger the scale. Scale={Large,Normal,Small},
TotalCapital	Total capital owned by the developer, Total capital=$\sum$profit-$\sum$cost+initial capital
InitialCapital	Developer-owned initial capital

the Attributes of Requirement Agent, we define the user's expected app service quality as UserExpect(asq), then the RequirementExpect = Funciton(UserExpect(asq)) = weight * UserExpect(asq), weight is a custom weight value.

C. Developer Agent

The Developer Agent is described by a set of attributes and behaviors, and the detailed description as show in Table IV and Table V.

D. Mobile App Agent

Mobile App Agent is described by a set of attributes, and the detailed description as show in Table VI.

E. Mobile Platform Agent

Mobile Platform Agent is described by a set behaviors, and the detailed description as show in Table VII.

IV. EXPERIMENT AND ANALYSIS

A. Experimental Environment

Repast Simphony, the complex system simulation platform [11] is used to build interaction models of the R-MSECO. We simulate the complex dynamic interactions of various entities in MSECO by adjusting the number of agents or setting different behavioral strategies of developers and platforms.

Table V
BEHAVIORS OF DEVELOPER AGENT

Attribute	Description
LearningBehavior	Determines the growth of skills, which require s a certain cost, it includes Self-learning, Inter active-learning and Social-learning
CheckRequirement	Check the user's requirements, the number of requirements that can be checked is determined by the developer's vision
DevelopBehavior	Check the market requirements and develop apps based on developers' own capabilities
MutualBehavior	Multiple developers use their strengths to collaborate on development to meet complex user needs
CombinedDevelop()	The expression function of developer cooperation. If a certain skill of developer A is greater than B, the skill of developer A is used, otherwise the skill of B is used.

Table VI
ATTRIBUTES OF THE MOBILE APP AGENT

Attribute	Description
PlatformData	PlatformData{DownloadNum, ReviewNum, Release Time}
DownloadNum	User download number
ComplaintRate	*ComplaintRate*=$\frac{ComplainNum}{DownloadNum}$, Refers to the ratio of app been complain
Cognition	*Cognition*= Function *F* (downloads, release time, developer size, platform promotion) *Cognition*=w1*DownloadNum+w2*ReleaseTime +w3*Scale+ w4*Promotion

Table VII
BEHAVIORS OF MOBILE PLATFORM AGENT

Attribute	Description
Routine Behaviour	Routine behavior of Mobile Platform, such as platform statistics
Promotion Behaviour	Promotional behavior can increase app Cognition
HQEA(High Quality Exclusive App)Strategy	Recommends high-quality exclusive apps (the platform judges the app service qual ity and uniqueness, and promotes it if it meets a certain conditions)
HQLC(High Quality Low Cognition)Strategy	Recommends the high-quality, low-cogni tion (less users) apps, (the platform judges the service quality and cognition of the app, and promotes it if a certain condit ions are met)
UserCF Strategy	Recommends apps that are of interest to most users in the user's group

B. Entity Behavior Strategy

Different entities and different behavior strategies will exert influence over other entities at the micro level and

the entire MSECO at the macro level in R-MSECO. In this work, we mainly focus on behavior strategies of developers and platform.

1) Developer strategy: In R-MSECO, the developer strategy is regarded as a learning behavior in the process of interaction between developers and requirements. The developers would gain new experience through learning and then continuously improve their develop ability. This learning behavior and can be divided into three types as below:

Self-learning: Developers improve their own development capabilities.

Interactive learning: Two developers learn from each other's strengths and improve their development capabilities.

Social learning: Learn the skills of developers with the most app downloads regularly.

2) Platform strategy: Platform strategy goal is to stabilize the entire MSECO, and make sure its prosperous and health. In R-MSECO, the platform strategy is mainly the promotion strategies. We divide promotion strategies into the following three types:

- The promotion strategy for high-quality and low-cognition apps (HQLC).
- The promotion strategy for high-quality and exclusive apps (HQEA).
- The group recommendation strategy (UserCF).

The commonality of the first two promotion strategy is to recommend high-quality apps, and their difference is that the HQLC focuses on that the apps with high service quality but without outstanding data cannot get the users' attention and lose the opportunity to be found by users. However, the HQEA strategy focuses on the competition between platforms.

C. Computing Environment Initialization

The initial settings of experimental parameters in R-MSECO are shown in Table VIII.

Table VIII
THE INITIAL SETTINGS OF EXPERIMENTAL PARAMETERS IN R-MSECO

Agent	Name	Setting
	Number	50
	Expect Service quality	Randomly generated between[0,10]
	Expect Platform data	Randomly generated between[0,10]
	Distribution	Randomly distribution
Developer	Number	25
	Energy	Randomly generated between[0,10]
	Cost	0
	TotalCapital	Normal distribution between(200,20)
	Distribution	Randomly distribution
App	Number	50
	Exclusive	Randomly generate 0 or 1
	ComplaintNum	0
	ServiceQuality	Randomly generated between[0,10]
	Distribution	Normal distribution
Requirement	Number	0
	RequirementExpect	0
Mobile Platform	TotalUsersNum	50
	TotalServicesNum	50
	TotalDevelopersNum	50

D. Experimental and Results

MSECO is a complex and dynamic ecosystem, in which various entities will continue to interact dynamically. We focus on how to make the entire MSECO healthy and sustainable. Therefore, our main concerns in this paper are as follows:

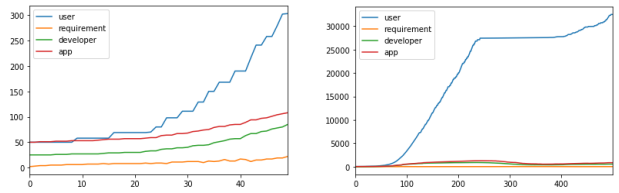
RQ1: What are the trends of the number of users, developers, apps, and requirements during the development of MSECO?

RQ2: What will happens of MSECO when the developer's external environment changes?

RQ3: How will developers' strategies affect MSECO?

RQ4: How will the platform's macro strategy affect MSECO?

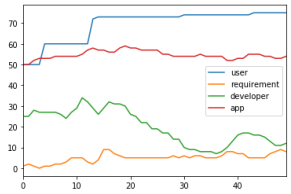
1) RQ1 Result: Figure 3(a) shows the change trend of each entity in the MSECO with the cycle. We can see that the users, developers, and apps have shown a linear growth as the cycle increases, among which the users has the fastest growth, and the requirements has remained low level, which means that the requirements generated during the user-app interaction will soon be identified by the developer and then complete the corresponding app update maintenance. In a real scenario, the user's requirements for the app are not all handled by the developer. When processing the user's requirements, the developer will consider both his own development capabilities and the development costs.



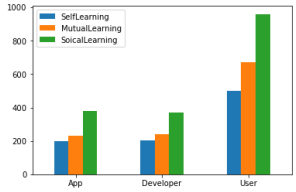
(a) The various entities changed in MSECO with the cycle (b) Variation trends of various entities in MSECO under long periods

Figure 3. RQ1 Results

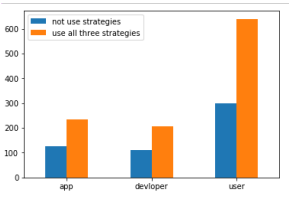
In addition, in theory, if all types of entities in MSECO do not adopt any special strategy, these entities will gradually



(a) Entity change trend in MSECO when developers' external environment changes



(b) Effects of different learning behaviors on various entities in MSECO



(c) Comparison of MSECO entities with and without three strategies

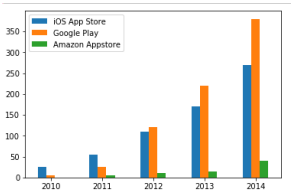
Figure 4. RQ2&RQ3&RQ4 Results

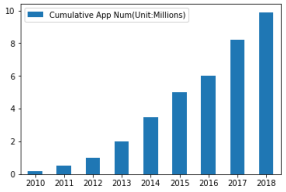
increase over time. Figure 3(b) shows the trends in users, developers, and requirements after 500 cycle iterations. We observe that after a sharp increase in users, there will be a period of slow growth, which is generally stepped. This indicates that MSECO will saturate at a certain critical point, and the growth will enter a certain flat period until the next large increase appears. The number of developers and apps will be relatively stable after a period of growth, and there will be no sharp increase or surge. We analyze the possible reason is that after MSECO has developed to a certain degree, its internal apps can basically meet the users' requirements and too many apps will make it difficult for users to find suitable apps.

2) RQ2 Result: The change of the developer's environment directly affects the speed and quality of app development and maintenance. Here, we mainly consider the development cost and consumption. Figure 4(a) shows the changes in MSECO when the developer consumption is increased and the initial development funds are small. We can find that developers and apps show ups and downs trend instead of linear growth. Meanwhile, the user growth has increased significantly in the early stage and then slowing down, and the requirements is much greater than the stable environment in RQ1. This is because the developer is the creator of the app in MSECO. When the developer group is affected, the maintenance of the app will be affected and the user experience will be affected also.

3) RQ3 Result: As described in Section IV(B), in order to improve the competitiveness in MSECO and obtain more benefits, the developers will adopt three behavior strategies, i.e., self-learning, interactive learning, and social learning. Figure 4(b) shows the overall trend of users, apps, and developers under three different learning behaviors. When developers adopt self-learning strategies, the growth rate of various entities in MSECO is the smallest, and the growth rate is the largest under the social learning mode. This shows that social learning is the best strategy to improve the market competitiveness for developers, which can provide a reference for new MSECO developers to improve their competitiveness.

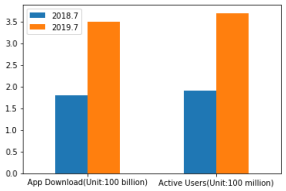
4) RQ4 Result: As a god's perspective in MSECO, the pursuit of platform manager is to further improve user satisfaction on the basis of meeting the users' basic requirement, rather than blindly pursuing the number of entities, so as to continuously attract users and increase the value of the entire MSECO. As described in Section IV(B), this article mainly considers three macro strategies of the platform, i.e., HQEA, HQLC and UserCF strategies. Figure 4(c) shows the comparison of the number of users, apps, and developers who used three strategies and without use 3 strategies after 50 cycles. It can be found that after using three strategies, the users, developers, and apps significant growth, which shows that these three strategies have practical value for the platform to attract users and develops.





(a) Changes in the number of developers in the three mobile application markets, 2010-2014(Unit: k)

(b) Changes in the number of apps in Googleplay



(c) Trends of app downloads and user changes in Huaweis application market

Figure 5. Case sduty results

E. Case Study

We analyze whether the results of our simulation are consistent with the real situation through the real MSECO, i.e., the historical data of the number of developers of Apple[1], Google[2], and Amazon App Market from 2010 to 2014. Figure 5(a) shows the changes in the number of developers in these three markets. We can find that the

[1]https://blog.appfigures.com/app-stores-growth-accelerates-in-2014/

[2]https://www.emarketer.com/chart/223442/total-apps-released-google-play-app-store-worldwide-2008-2018-millions

number of developers in the mobile app market has shown a rapid growth trend, which validates the results of our experiment in RQ1. Meanwhile, Figure 5(b) shows that the number of apps in the Google market from 2008 to 2018 has shown explosive growth, which also shows the consistent with our RQ1 experiment results.

Regarding the platform's macro strategy, we take Huawei's global application market as an example. This market attaches great importance to the UserCF Strategy. Through global app classification, local/nearby app recommendation, model adaptation app recommendation, and synergy to strengthen the effect of the same group recommendation. Figure 5(c) shows the trend of app downloads and user changes after Huawei's app market adopts the UserCF Strategy. We can see that the number of app downloads and active users has increased significantly, which verifies the correctness of our platform macro strategy.

V. Related Work

Software Ecosystem (SECO) represent a new approach for software product development. Bosch et al. [12] think that SECO includes software digital platform, internal and external developers, a set of business rules that define and regulate how the parts relate to one another, and the actors involved in the process and other roles to complete the operation of SECO. In the field of mobile application platform, such as Google play, it is called mobile software ecosystem, i.e., MSECO.

Awdren Fonto et al. [1] divide MSECO roles into platform, user, developer, community, mobile app, mobile app store and evangelist entity roles. In fact, the mobile software ecosystem is a special SECO in essence, which focuses on the mobile application market. The mobile app store is the direct embodiment of MSECO. Currently, the research on MSECO is still in the initial stage [3], which mainly analyzes the entity roles in MSECO from the conceptual level, such as focus on developer [4][5], platform working mode and empirical analysis of relevant data around MSECO.

Around the developers in MSECO, Fonto A et al. [13] studied the impact of the developer experience in MSECO, and analyzed how to manage and evaluate the developers in MSECO from the perspective of the repository. Goldbach T et al. [6] discussed the impact of control theory on developer performance, and provide platform owners with insights on the key role of developer perceived autonomy in influencing their performance and loyalty. Alsubaihin A et al. [14] studied the app store from the perspective of developers to investigate the impact of the app store on software engineering tasks. Wang H et al. [15] conducts exploratory research on the mobile application ecosystem from the perspective of developers, and analyzes the motivation behind different behaviors of developers, and they believe that understanding the behaviors of developers is not only helpful to other developers, but also for the market and users.

According to the working mode of mobile platform in MSECO, Fonto A et al. [9] pointed out that developers have diversity to perform marketing activities, as well as to find materials that support development. Rietveld J et al. [16] discovery platform is not simply to promote the app products under the platform of "best in group", through the research on the promotion strategy of video game platform market, this paper discusses the complex strategic choices made by platform sponsors when using selective promotion to manage the ecosystem. Fontao A et al. [17] mainly discusses how the platform supports developers and how to extract the influencing factors of developers' experience, and puts forward a process to support mobile app developers to build app in MSECO.

On the other hand, empirical analysis of real data in MSECO is also a hot research trend. Pagano D et al. [8] analyzed more than one million comments from Apple app store, and investigated the way and time which users provided feedback, and they found that most of the feedback was provided shortly after the release of the new version. Finkelstein A et al. [18]use app store analysis to understand the rich interaction between app customers and developers. Li H et al. [19] analyze the big data of Wandoujia to investigate how the choice of device model affects user behavior. Liu X et al. [20] conducts empirical research on behavioral service profiles collected from millions of users in Wandoujia, and understands different usage patterns of smartphone from large-scale app store service profiles.

MSECO is a complex and dynamic ecosystem, and simulation modeling is the most reasonable way to analysis complex and dynamic interaction in MSECO. To best of our knowledge, there is rarely work on multi-agent simulation analysis of MSECO, only Lim et al. [21] have simple simulation on developers and app entities in MSECO, they studied the developer's strategy, the spread of the app, and the impact of the app ranking algorithm respectively, and the results show the necessity and feasibility of simulation model.

However, the focus of above works still on the impact of single entity on another entity in MSECO, or simply analyzing one side of MSECO, but there is no macro description of the internal comprehensive dynamic interaction mechanism of all entities in MSECO. In fact, due to the complexity of MSECO, it is full of challenges to characterize the internal operation mechanism of MSECO and study on its internal operation mechanism is of great significance for future MSECO research.

VI. Conclusions and Future Work

MSECO is a complex and dynamic ecosystem that integrates multiple entity elements. In this article, we propose a user requirement-driven MSECO operating model R-MSECO which include users, developer, app, requirement, and mobile platform. Then, we simulated the results of

various entity interactions based on multi-agent simulation technology, and finally verified the validity and feasibility of R-MSECO using real mobile app store data. Our experimental results can provide support for different entity behavior decisions in MSECO. In addition, it can also help mobile platform owners in operating and formulating future strategies for the platform. Future research will focus on the multi-party game mechanism between the user group, the developer group, and the platform, and jointly describe the internal evolution mechanism of MSECO.

ACKNOWLEDGMENT

This work is supported by the National Key R&D Program of China grant No.2017YFB1401201, the National Natural Science Key Foundation of China grant No.61832014 and National Natural Science Foundation of China grant No.61972276, the Shenzhen Science and Technology Foundation grant JCYJ20170816093943197, and the Natural Science Foundation of Tianjin City grant No.19JCQNJC00200.

REFERENCES

[1] A. de Lima Fontão, R. P. dos Santos, and A. C. Dias-Neto, "Mobile software ecosystem (mseco): a systematic mapping study," in *2015 IEEE 39th Annual Computer Software and Applications Conference*, vol. 2. IEEE, 2015, pp. 653–658.

[2] S. Jansen and E. Bloemendal, "Defining app stores: The role of curated marketplaces in software ecosystems," in *International Conference of Software Business*. Springer, 2013, pp. 195–206.

[3] C. Steglich, S. Marczak, L. P. Guerra, L. H. Mosmann, M. Perin, F. Figueira Filho, and C. de Souza, "Revisiting the mobile software ecosystems literature," in *2019 IEEE/ACM 7th International Workshop on Software Engineering for Systems-of-Systems (SESoS) and 13th Workshop on Distributed Software Development, Software Ecosystems and Systems-of-Systems (WDES)*. IEEE, 2019, pp. 50–57.

[4] A. Fontão, O. M. Ekwoge, R. Santos, and A. C. Dias-Neto, "Facing up the primary emotions in mobile software ecosystems from developer experience," in *Proceedings of the 2nd Workshop on Social, Human, and Economic Aspects of Software*, 2017, pp. 5–11.

[5] A. Fontão, F. Lima, B. Ábia, R. P. dos Santos, and A. C. Dias-Neto, "Hearing the voice of developers in mobile software ecosystems," in *Proceedings of the 31st Brazilian Symposium on Software Engineering*, 2017, pp. 4–13.

[6] T. Goldbach, A. Benlian, and P. Buxmann, "Differential effects of formal and self-control in mobile platform ecosystems: Multi-method findings on third-party developers continuance intentions and application quality," *Information & Management*, vol. 55, no. 3, pp. 271–284, 2018.

[7] A. de Lima Fontão, R. P. dos Santos, and A. C. Dias-Neto, "Exploiting repositories in mobile software ecosystems from a governance perspective," *Information Systems Frontiers*, vol. 21, no. 1, pp. 143–161, 2019.

[8] D. Pagano and W. Maalej, "User feedback in the appstore: An empirical study," in *2013 21st IEEE international requirements engineering conference (RE)*. IEEE, 2013, pp. 125–134.

[9] A. Fontão, R. Santos, A. Dias-Neto *et al.*, "Mseco-dev: Application development process in mobile software ecosystems," in *Proceedings of the International Conference on Software Engineering and Knowledge Engineering*, 2016, pp. 317–322.

[10] S. Mukhopadhyay, M. de Reuver, and H. Bouwman, "Effectiveness of control mechanisms in mobile platform ecosystem," *Telematics and Informatics*, vol. 33, no. 3, pp. 848–859, 2016.

[11] S. L. Lim, P. J. Bentley, N. Kanakam, F. Ishikawa, and S. Honiden, "Investigating country differences in mobile app user behavior and challenges for software engineering," *IEEE Transactions on Software Engineering*, vol. 41, no. 1, pp. 40–64, 2015.

[12] J. Bosch and P. Bosch-Sijtsema, "From integration to composition: On the impact of software product lines, global development and ecosystems," *Journal of Systems and Software*, vol. 83, no. 1, pp. 67–76, 2010.

[13] A. D. L. Fonto, A. C. Dias-Neto, and R. P. D. Santos, "Towards a guideline-based approach to govern developers in mobile software ecosystems," in *16th International Conference on Software Reuse (ICSR 2017)*, 2017.

[14] A. AlSubaihin, F. Sarro, S. Black, L. Capra, and M. Harman, "App store effects on software engineering practices," *IEEE Transactions on Software Engineering*, pp. 1–1.

[15] H. Wang, L. Zhe, G. Yao, X. Chen, Z. Miao, G. Xu, and J. Hong, "An explorative study of the mobile app ecosystem from app developers' perspective," 2017.

[16] J. Rietveld, M. A. Schilling, and C. Bellavitis, "Platform strategy: Managing ecosystem value through selective promotion of complements," *Organization Science*, vol. 30, no. 6, pp. 1232–1251, 2019.

[17] A. D. L. Fontão, R. P. Dos Santos, and A. C. D. Neto, "Mseco-sup: Support process in mobile software ecosystems," in *2015 29th Brazilian Symposium on Software Engineering*. IEEE, 2015, pp. 31–40.

[18] A. Finkelstein, M. Harman, Y. Jia, W. Martin, F. Sarro, and Y. Zhang, "Investigating the relationship between price, rating, and popularity in the blackberry world app store," *Information and Software Technology*, vol. 87, pp. 119–139, 2017.

[19] H. Li and X. Lu, "Mining device-specific apps usage patterns from large-scale android users," *arXiv preprint arXiv:1707.09252*, 2017.

[20] X. Liu, H. Li, X. Lu, T. Xie, Q. Mei, F. Feng, and H. Mei, "Understanding diverse usage patterns from large-scale appstore-service profiles," *IEEE Transactions on Software Engineering*, vol. 44, no. 4, pp. 384–411, 2017.

[21] S. L. Lim and P. J. Bentley, "How to be a successful app developer: Lessons from the simulation of an app ecosystem," *ACM SIGEVOlution*, vol. 6, no. 1, pp. 2–15, 2012.