Techniques for Modeling Mobile Application Spreading

Ádám Horváth, and Károly Farkas, *Member, IEEE*

***Abstract*—While the number of mobile devices and applications increases considerably, application spreading by using direct communication between mobile devices has not got too much attention so far. However, exploiting the advantages of modern communication paradigms has even economic relevance. For instance, direct communication in ad hoc networks can be used to advertise and spread applications, which influences the number of purchases what application providers may want to estimate.**

In this paper, we present two main techniques for modeling application spreading in this environment. First, we model the spreading process using Closed Queuing Networks, assuming a single type user behavior. Then, to capture different user behaviors, we show how to model application spreading by Stochastic Petri Nets. We define a basic Petri net model that we analyze using a mean field based methodology. Unfortunately, introducing more realistic user behaviors makes the analytical handling of the Petri net too complex, or even impossible. We show an example of this case extending the basic Petri net model with an additional, more realistic user behavior, and investigate this extended model via simulations. Moreover, we compare the investigation results of the different models and point out their relations.

***Index Terms*—Application Spreading, Closed Queuing Networks, Mathematical Modeling, Stochastic Petri Nets, Mean Field Based Methodology**

I. INTRODUCTION

With the proliferation of mobile devices, the number of mobile applications is increasing significantly. However, mobile application spreading is not a frequently investigated research topic today. Being aware of the characteristics of application spreading is important for the application provider not only from technical, but also from economic point of view. The application provider has to know or at least assess how much money he can earn from the purchases of a given application; how much time is needed to realize it; and which factors influence the spreading process and how.

Traditionally, mobile applications are spread via a central entity, like an Internet webshop. Users can browse the website of the merchant, select, purchase and download the application they like. However, decentralized technologies such as self-organized or mobile ad hoc networks allow the users to get the application software directly from each other. Hence, direct application download can change the characteristics of traditional application spreading. The participants of this direct communication can even try out the applications and be motivated to purchase the ones they liked (purchasing is available only via a traditional way, because secure payment in this environment is still a challenging issue today).

In this type of communication, a lot of factors can change the characteristics of the spreading process, especially from economic viewpoint, which have not taken into consideration yet (e.g., community experience when playing a multi-player game). These factors can give more motivation to the users to purchase the application than they would have seen only some advertisements.

In this paper, we present two modeling techniques by which we can investigate application spreading in mobile environment. First, we propose the use of Closed Queuing Networks (CQNs) [1] for modeling the application spreading process. Assuming a single, homogeneous user behavior, we can simply and quickly obtain the expected number of purchases by using a CQN.

Then, we propose Stochastic Petri Nets (SPNs) [2] based modeling to capture more sophisticated user behaviors. We define first a basic SPN model assuming different user types. We analyze this model using a fluid approximation method, originally presented in [3]. In this method, we transform the SPN into ordinary differential equations (ODEs), which can be evaluated quickly even assuming a huge user population. Unfortunately, if we want to include more realistic user behaviors in the model, the analytical handling of the Petri net can become too complex, or even impossible. We show an example of this case extending the basic Petri net model with an additional, more realistic user behavior, and investigate this extended model via transient simulation. However, we have to keep in mind that simulating the transient behavior of Petri nets with large state space is much slower and performance intensive task than analytical evaluation.

Finally, we compare the investigation results of the different models and point out their relations.

The rest of the paper is organized as follows. In Section II, we present a short overview of the related works. In Section III, we describe the communication model and define the user behavior types. We present two techniques to investigate the application spreading process, thus our CQN and SPN models together with some results in Section IV and Section V, respectively. Finally, we compare the results and conclude the paper in Section VI.

Manuscript received February 29, revised April 23, accepted for publication April 27, 2012.

Á. Horváth is with the Institute of Informatics and Economics, University of West Hungary (UWH), Sopron, Hungary (corresponding author, phone: +36-99-518-606; fax: +36-99-518-367; e-mail: horvath@inf.nyme.hu).

K. Farkas is with the Department of Telecommunications, Budapest University of Technology and Economics (BME), Budapest, Hungary (e-mail: farkask@hit.bme.hu).

II. Related Works

With the proliferation of modern communication paradigms, the investigation of application spreading using new ways becomes more and more important. However, it has not got too much attention so far. Besides our previous contributions [4]-[6], only a few papers touch even the commercial use of ad hoc networks and direct communication.

On the other hand, epidemic spreading is a popular research topic today and this area is similar to our context. In [7], the authors present a model, by which they investigate the propagation of a virus in a real network. In [8], the authors present scale-free networks for modeling the spreading of computer viruses and also give an epidemic threshold, which is an infection rate. Information spreading is also investigated by using epidemic spreading models, such as the susceptible-infected-resistant (SIR) model [9], or other models based on the network topology [10], [11]. In [12], malicious software spreading over mobile ad hoc networks is investigated. The authors propose the use of the susceptible-infected-susceptible (SIS) model based on the theory of Closed Queuing Networks.

In [13], the authors propose the commercial use of ad hoc networks and present a radio dispatch system using mobile ad hoc communication. In the proposed system, the connectivity of the nodes is the key element of information dissemination. In our models, we do not consider the network topology as a key element of application spreading, since no real-time information dissemination is needed between the users. For the same reason, we do not deal with mobility models such as random walk model, which are well presented in many contributions [14]-[16].

Although the above mentioned proposals show some similarities with our work, none of them deals with application spreading and, except [13], they do not touch the commercial benefits of direct communication. Moreover, the authors in [13] consider information dissemination as a tool, and not as a goal.

III. Communication Model and User Types

In this section, we present the communication model which we use in our investigations. Moreover, we introduce three user types based on different user behaviors.

A. Communication Model

We refer to the individuals who are interested in the use of the application as users. The population that we investigate is composed of users only, and we do not take uninterested users into consideration, because they do not influence the spreading process. Therefore, we assume a closed user population.

We investigate the spreading of a given multi-user application having two versions, a trial and a full version. The users can be categorized into different classes depending on whether they do possess any version of the given application or do not. We named the classes after the terminology of epidemics, since our model shows similarity to the epidemic spreading models. A user is called (1) infected, if he has got the full version of the application; (2) susceptible, if he possesses only the trial version of the application; and (3) resistant, if he has got none of them, or he has already lost the interest of using the application.

Users with their mobile devices form self-organized networks from time to time, in which direct communication takes place. The trial version of the application is free and available in these networks, so users can download it and even try it out. However, it has some restrictions (see later), so the users have to purchase the application for unrestricted usage via a traditional way of purchasing. Later, also these users can spread the trial version of the purchased application further.

Since we want susceptible users to be motivated in purchasing the full version of the application, some limitations must be made in using the trial version. Therefore, we apply a limit (leech[1] limit) that restricts how many nodes possessing the trial version (leech) can connect to a node possessing the full version (seed[1]). In this sense, the seeds can be considered as servers, which can serve a limited number of clients. A seed is always an infected user, while a leech may be either infected or susceptible. Fig. 1 depicts the case when a susceptible user purchases the application.

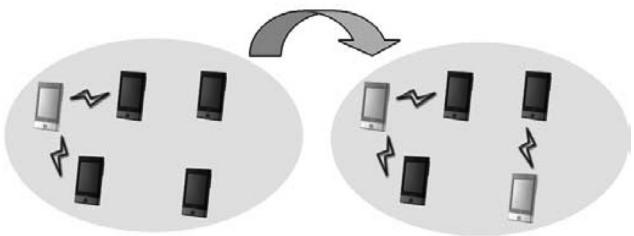

Fig. 1. Change of application usage when a susceptible user purchases the application.

The devices form an ad hoc network, in which the dark devices depict susceptible users, while the light ones depict infected users. In this example, the leech limit is two, so two susceptible users can peer to the only infected user, while the other two have to wait (the connection symbol represents application level peering). After one of them purchased the application, they can also use it, as shown in the right side of Fig. 1.

B. User Types

Beyond the basic communications, we distinguish three different user types based on the users' behavior. $Type_A$ users are interested in using the given application, therefore, they are its potential buyers even without trying it out. $Type_B$ users are motivated in purchasing the application only if they realize that the application is popular in their environment. $Type_C$ users also purchase the application very likely, but they will do it with a given intensity, only if they cannot find a seed from time to time which they can connect to.

In our CQN model, we use only $Type_A$ users, since we additionally introduce the other user behaviors in our SPN models, namely $Type_B$ users in the basic and $Type_B$, $Type_C$ users in the extended SPN model. Of course, additional user types can be introduced, too. However, the more user types we capture the more complex model we get, which can make the handling of the model difficult.

[1] After the terminology of BitTorrent [17].

IV. MODELING WITH CLOSED QUEUING NETWORKS

In self-organized networks, where spontaneous communication takes place, the network topology can change rapidly due to the high degree of mobility. These topology changes can be modeled by stochastic processes [18]. We can appropriately describe a stochastic process in a closed population, which is interesting from our point of view, using Closed Queuing Networks [1]. Moreover, ordering transition intensities to the state changes we can capture the time behavior of the application spreading process, too.

A. Spreading Model

Assuming a homogeneous user population with simple user behavior (only Type_A users) we propose the CQN depicted in Fig. 2 to model the application spreading process. CQN models are not appropriate to handle complex conditions, but they can be used as a first approach in simple situations providing quick results.

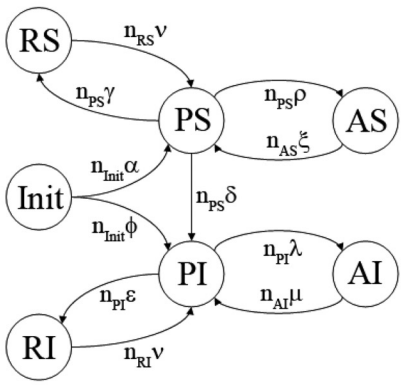

Fig. 2. The proposed CQN model.

The users are represented by the different model states in Fig. 2. Each user is in a given state depending on his current user class. The resistant users are in state *Init*. They possess neither the trial nor the full version of the application. We call also resistant the users, who have already lost the interest in using the application, however, they possess either its trial (state *RS*) or its full version (state *RI*). The susceptible users are in state *PS* and *AS*, depending on that they are currently using the application (Active Susceptibles, *AS*) or not (Passive Susceptibles, *PS*). Similarly, the active and passive infected users are in state *PI* and *AI*, respectively.

The Greek letters in Fig. 2 denote transition intensities regarding to a single user. The transition intensity is a real number illustrating how many times a transition expectedly takes place in a given time interval. n_x represents the number of users in state x, so the transition intensity of a given state transition is proportional to the number of users in the source state. The state transitions of the model are described in Table I.

B. Usage of the Spreading Model

We can unambiguously describe the state of the system with the user distribution (n_{Init}, n_{PS}, n_{AS}, n_{PI}, n_{AI}, n_{RS}, n_{RI}). The transition intensities (α, γ, δ, ε, ϕ, λ, μ, ν, ρ and ξ) regarding to a single user are the system parameters, which are hard to be determined theoretically. In this paper, we set the system parameters based on common sense. The parameter setting can be fine-tuned experimentally, what is beyond the scope of this paper.

TABLE I
STATE TRANSITIONS OF THE PROPOSED CQN MODEL

Transitions	Description
PS → AS	A susceptible user starts to run the application and tries to connect to a seed in the network. If he cannot find one, he has to wait.
AS → PS	A susceptible user stops running the application.
PI → AI	An infected user starts to run the application, then either he tries to connect to a seed, or will be a seed himself to which leeches can connect.
AI → PI	An infected user stops running the application.
Init → PS	A resistant user downloads the trial version of the application.
PI → RI	An infected user becomes resistant losing the interest in using the application.
PS → RS	A susceptible user becomes resistant losing the interest in using the application.
Init → PI	A resistant user purchased the application without trying it out.
PS → PI	A susceptible user becomes infected by purchasing the application.
RS → PS	It is possible that a resistant user, who lost the interest in using the trial version, wants to use the application again after a while. If so, his state becomes susceptible again.
RI → PI	Similarly, if a resistant user possessing the full version of the application wants to use it again, his state changes to infected.

In each system state, we can generate the holding time h (the time that the system is expected to spend in a given system state) as an exponentially distributed random variable[2] in the following way:

$$h = \frac{-lnRND}{\sum_{\forall state\ x} out_x} \tag{1}$$

where $0 \leq RND < 1$ is a pseudo-random number, and out_x denotes the sum of the intensities for each transition with source state x. We must compute h after each state change, because the user distribution changes when a transition takes place. We can generate the next system state based on the ratio of the current transition values. After we generated the transition that takes place, we move one user from its source to its destination state, and compute the holding time of the new system state, and so on.

At the beginning, each user is in state *Init*, which is the initial state of the system. The users will leave this state and change their states from time to time. After a while, a user will lose the interest in the application usage (reaches state *RS* or *RI*), but it does not mean that he cannot be interested again later on. Thus, the state transitions $RS \rightarrow PS$ and $RI \rightarrow PI$ are also enabled, however, we allow them only with low intensity values. Therefore, we will reach a system state (final system state) sooner or later, in which each user is either in state *RS* or

[2] Using exponentially distributed holding times is a usual modeling simplification in this context [12], [19].

PS. The final system state is not the steady state, however, our investigation will stop here. Taking into account the asymmetry of this system (if someone purchases the application, he will never lose it), each user will be in one of state *RI*, *PI* or *AI* by reaching the steady state (even product-form solution exists). However, it is meaningless to consider this state in the model investigation, since after reaching the final system state the holding times become extremely large, so the system changes very slowly. Hence, our investigations are always transient and consider only a time period which is interesting from the merchant's point of view.

We can determine how many pieces of the application were sold, upon reaching the final state, by summing the number of $PS \rightarrow PI$ and $Init \rightarrow PI$ transitions. Running simulations and evaluating the results, we can see the time characteristics of the spreading process, too.

C. Results Derived from CQN

In this section, we recall some analytical results from [5] and validate them via simulations. In contrast to [5], we use a single user type in this model, and apply such a parameter setting which makes possible to compare the results to the SPN models.

Analytical solutions of CQNs can be obtained, e.g., with the well-known Mean Value Analysis (MVA) [20]. However, we investigate the transient behavior of the system, in which case this method is not feasible.

After a while, each user will leave the initial state, since we do not take the uninterested individuals into consideration. Based on the intensity value of transition $Init \rightarrow PI$ and $Init \rightarrow PS$, we can determine how many users will expectedly purchase the application without trying it out (direct purchases, *DP*) in the following way:

$$DP = n_{Init} \cdot \frac{\phi}{\alpha + \phi} \tag{2}$$

where n_{Init} denotes the initial number of users in state *Init*, i.e., the total population size. All nodes (users) that did not purchase the application without trying it out will change their state to susceptible. Susceptible states are state *PS* and *AS*, and there are two possibilities for the users to leave these states: a user can become either (1) resistant (transition $PS \rightarrow RS$); or (2) infected (transition $PS \rightarrow PI$). Moreover, it is also allowed to return from state *RS*, but the intensity of transition $RS \rightarrow PS$ is very low, since losing the interest and being interested again after a while is not a typical user behavior. Therefore, we can estimate the number of indirect purchases (purchase after trying out) based on the ratio of transition $PS \rightarrow PI$ and $PS \rightarrow RS$ as follows:

$$IDP = n_{Init} \cdot \frac{\alpha}{\alpha + \phi} \cdot \frac{\delta}{\delta + \gamma} \tag{3}$$

The expected value of total purchases (*TP*) can be obtained by summing (2) and (3).

To validate the analytical results we ran simulations. Table II compares the analytical results to the average of

TABLE II
COMPARISON OF THE ANALYTICAL AND SIMULATION RESULTS USING THE CQN MODEL

Description	Analytically	By simulation
DP	4.95	4.95
IDP	165.02	164.99
TP	169.97	169.94

10000 individual simulation runs with the following parameters: $n_{Init} = 500$, $\alpha = 10^{-3}$, $\gamma = 210^{-3}$, $\delta = 10^{-3}$, $\varphi = 10^{-5}$. The other parameters do not influence *DP* and *IDP*, however, they have an effect on the time behavior of the spreading process. This can be investigated also via simulations.

V. MODELING WITH STOCHASTIC PETRI NETS

In this section, after introducing the fundamentals of Stochastic Petri Nets [2], to be able to handle also $Type_B$ and $Type_C$ users we present the description and the analysis of our two SPN models, the basic and the extended one.

Since $Type_A$ users are present in all of our models (including the CQN model), we can compare the models from the viewpoint of this simple user behavior, see Section VI.

A. SPN Formalism

The continuous-time Stochastic Petri Net can be defined as a 6-tuple

$$(P, T, I, O, \lambda, M_0)$$

where $P = \{p_i\}$ is the set of *places*; $T = \{t_i\}$ is the set of *transitions*; $I, O : T \times P \rightarrow \mathbb{N}$ are the *input* and *output* functions that define the arcs of the net with their multiplicities; $\lambda : T \rightarrow \mathbb{R}$ are the functions that assign *firing intensities* to each transition and M_0 is the *initial marking* of the net.

A transition t is enabled in marking M if $M(p) \geq I(t, p)$ holds for all places p. In other words, a transition is enabled if each of its input places contains at least the amount of tokens defined by the input function I. Only enabled transitions can fire, and if one does, we remove tokens from the input places and add tokens to the output places of the firing transition. Formally, the new marking M' is given as $M'(p) = M(p) + O(t, p) - I(t, p)$ for all $p \in P$.

The transitions fire after a random delay that can be described with an exponentially distributed random variable. Its parameter depends on the given transition's firing intensity and the current marking. Our model uses the infinite server approach, thus the firing intensity is increasing with the increase of the tokens' number in the enabling places. This concept is formally captured by the definition of enabling degree. Namely, the enabling degree $ed(t, M)$ of a transition t in the marking M is d iff $\forall p \in P$, $M(p) \geq dI(t, p)$ and $\exists p \in P$: $M(p) < (d + 1) I(t, p)$.

For further details on SPNs, see [2] or [21].

B. Basic SPN Model

Here we describe our basic SPN model, and present a mean field based methodology for analyzing SPNs. Then using this

methodology, we show the transient analysis of our basic SPN model.

Model Description

Our basic SPN model is depicted in Fig. 3. The rectangles illustrate the transitions of the SPN, while the circles represent the places. We assume the presence of 500 $Type_A$ and 500 $Type_B$ users, and initially each user is in the passive susceptible states *PASS_A* ($Type_A$ users) and *PASS_B* ($Type_B$ users). Seeds can also be passive (*PASS_S*), however, there are no seeds in the network initially. Similarly to the CQN model, we call a user active if he is currently using the application. We keep count of the number of active $Type_A$ users (*ACT_A*), $Type_B$ users (*ACT_B*), available seeds (*FREE_S*) and the total number of active users, including seeds (*ACT_USERS*). Since the leech limit is one, the available seeds show how many susceptible users can start the application in a given time. We also keep count of the number of application purchases either by $Type_A$ users (*PURCHASES_A*) or by $Type_B$ users (*PURCHASES_B*). After a while, users lose the interest in using the application in this model, as well. If so, they change their state to resistant (*LOST_INT_A*, *LOST_INT_B*, *LOST_INT_S*).

The transitions of the model and their values regarding to one user with which we analyzed the net are described in Table III.

TABLE III
STATE TRANSITIONS OF THE BASIC SPN MODEL

Transitions	Description
START_A / START_B / START_S	A $Type_A$ user / $Type_B$ user / seed starts to run the application. They can fire only if there is at least one available seed in the network ($4 \cdot 10^{-2}$ in each case).
STOP_A / STOP_B / STOP_S	A $Type_A$ user / $Type_B$ user / free seed stops running the application ($9 \cdot 10^{-1}$ in each case).
SEED_DISC_A / SEED_DISC_B	A seed to which a $Type_A$ / $Type_B$ user had been connected stopped running the application. The state of the connected leech node becomes passive ($9 \cdot 10^{-1}$ in each case).
LOSE_INT_A / LOSE_INT_B / LOSE_INT_S	A $Type_A$ user / $Type_B$ user / seed loses the interest in using the application (10^{-3} in case of a seed, $2 \cdot 10^{-3}$ otherwise).
PURCHASE_A / PURCHASE_B	A $Type_A$ user / $Type_B$ user purchases the application (10^{-3} in each case).
INT_AGAIN_A / INT_AGAIN_B / INT_AGAIN_S	It is possible that a resistant user, who lost the interest in using the application, wants to use the application again after a while. If so, his state becomes susceptible or infected again, depending on his previous state (10^{-5} in each case).

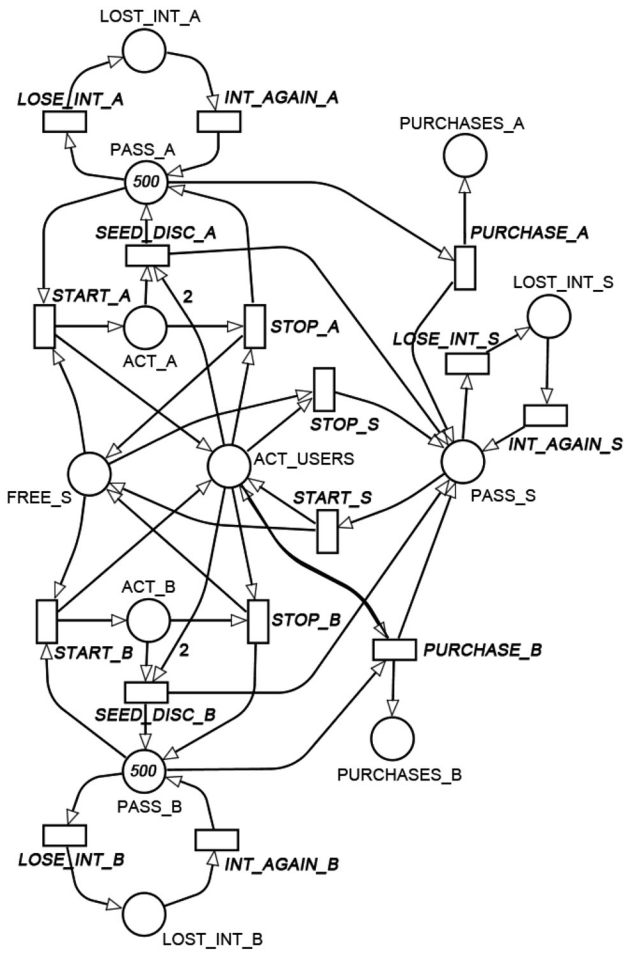

Fig. 3. The basic SPN model.

Our basic SPN model does not contain inhibitor arcs, thus we can apply the mean field based methodology in its analysis. If we have to build a more complex model enabling also the use of inhibitor arcs, its analytical handling is not possible anymore, so we must run simulations to investigate the model's behavior (see Section V.C).

Mean Field Based Methodology

The standard approach for analyzing SPNs is to construct the continuous time Markov chain (CTMC, for details see, e.g., [22]) corresponding to the underlying stochastic behavior of the SPN and perform the steady state or transient analysis analytically [23] or by simulation. However, this approach becomes unfeasible due to the size of the state space if we consider a network composed of a large number of mobile components.

In the following, we describe the mean field approach, which is a fluid approximation method for model evaluation. Applying this method, the analysis will terminate within a few seconds, even when the state space explodes due to the high number of tokens. The following definition and theorem are based on [3], while we presented it in [6] in a form that is directly related to the applied definition of SPN. In that paper, we provided a formal relation between the CTMC and its fluid approximation, too.

Definition: A parametric family of Markov chains, $X_\upsilon(t)$ with $\upsilon \in \mathbb{N}$, with state spaces $E_\upsilon \subset \mathbb{Z}^k$, is called density dependent if and only if there exists a continuous function $f(x, l)$, $x \in \mathbb{R}^k$, $l \in \{L(t_1), \ldots, L(t_m)\}$, such that the non-diagonal entries of the infinitesimal generator corresponding to $X_\upsilon(t)$ can be written in the following form:

$$q_{k,k+l} = \upsilon f\left(\frac{k}{\upsilon}, l\right), l \in \{L(t_1), \ldots, L(t_m)\} \quad (4)$$

and the initial state of the chain is υx_0, $x_0 \in \mathbb{Z}^k$, with probability 1. Let $X(t)$ denote the solution of the ODEs:

$$\frac{dX(t)}{dt} = \sum_{l \in \{L(t_1),\ldots,L(t_m)\}} f\,(X(t), l) \quad (5)$$

with initial condition $X(0) = x_0$.

Theorem: Under mild conditions of function f (for details see [3]), the following relation holds between function $X(t)$ and a trajectory of the CTMC $X_\upsilon(t)$:

$$\forall \delta > 0: \lim_{\upsilon \to \infty} P\left\{sup_{s \le t}\left|\frac{1}{\upsilon} X_\upsilon(s) - X(s)\right| > \delta\right\} = 0 \quad (6)$$

The interpretation of (6) is the following. Consider a CTMC modeling the interaction of k quantities with $\mathbb{Z}^k$ state space. If we observe a sequence of CTMCs with increasing initial state, and this increase gives rise to a sequence of infinitesimal generators corresponding to the form in (4), then as υ is increased, the behavior of the CTMC converges to the solution of the ODEs in (5). It means that the probability of finding any difference between the trajectory of the CTMC and the solution of the ODEs in a finite time horizon $(0, t)$ is zero.

It has already been shown in [3], that for large population sizes, the corresponding ODEs provide a good approximation of the system's behavior in case of density dependent CTMCs. It is straightforward to show that the basic Petri net we use for modeling application spreading is density dependent, therefore, we can approximate the behavior of this Petri net with high number of tokens by solving the ODEs. Moreover, we presented in [6] that the approximation works even with a lower number of tokens.

Transient Analysis

In the following, we illustrate the usage of the mean field approach through the transient analysis of the above mentioned basic SPN model.

The solution of the ODEs is a good approximation of the average behavior of the model. Fig. 4 depicts the cumulative expected value of the number of $Type_A$ and $Type_B$ users' purchases after a given time, while Fig. 5 shows the cumulative expected value of the number of users who lost the interest in using the application as time elapses.

In Fig. 4 and Fig. 5, we can see that mostly $Type_A$ users purchased the application, and the interest in using the application is approximately limited to the first 4000 hours. However, the parameter settings are critical, since the number of $Type_B$ users' purchases depends on the activity of the users. The higher the activity is, the more $Type_B$ users will purchase the application.

C. Extended SPN Model

In this section, we describe our extended SPN model and its transient simulation.

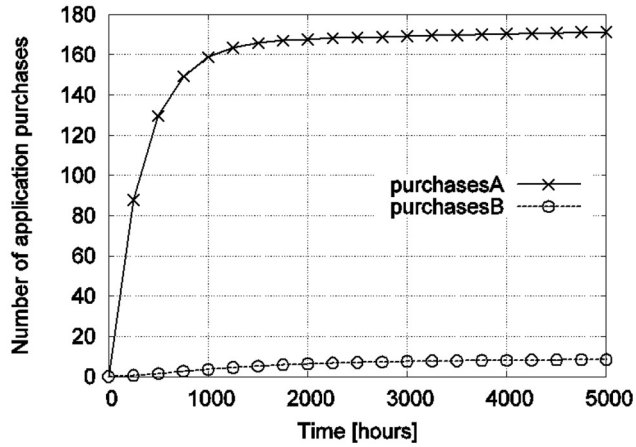

Fig. 4. Expected value of the number of application purchases as the function of elapsed time.

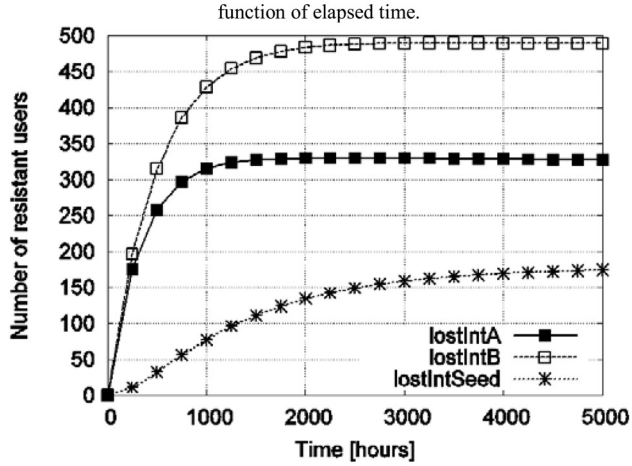

Fig. 5. Expected value of the number of users who lost the interest in using the application.

Model Description

In our extended SPN model, we can also handle more sophisticated user behaviors, like the third user type ($Type_C$). This requires the use of inhibitor arcs, too.

Apart from this, the extended model is very similar to the previous one. To handle the new user type we added four places (*PASS_C*, *ACT_C*, *LOST_INT_C* and *PURCHASES_C*) and six transitions (*LOSE_INT_C*, *INT_AGAIN_C*, *PURCHASE_C*, *SEED_DISC_C*, *START_C* and *STOP_C*) to the basic model. As we described in Section III.B, $Type_C$ users purchase the application only if they cannot find an available seed which they can connect to. Therefore, the transition *PURCHASE_C* is enabled only if there is no token in place *FREE_S*. This relationship is denoted in the model by an inhibitor arc with a circle on its head. Beyond that, $Type_C$ users' behavior is similar to the others'.

Fig. 6 shows our extended SPN model. The newly added part is marked by grey background. As earlier, we set 500 users from each user type in the initial marking.

Transient Simulation

Since our extended SPN model contains also inhibitor arc, we cannot use the fluid approximation method to investigate the model's behavior, rather we have to run simulations. There exist many tools for modeling with SPNs, e.g., the ones

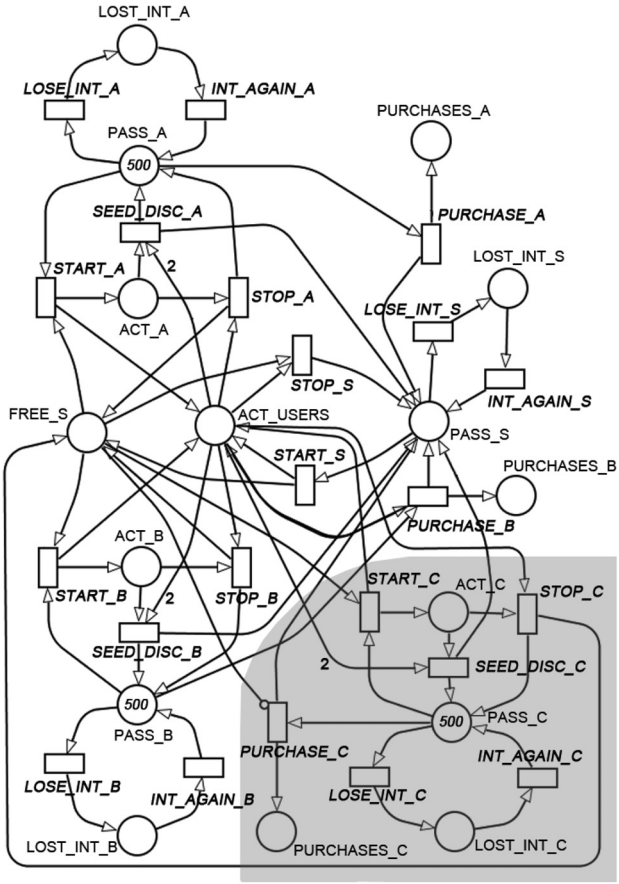

Fig. 6. The extended SPN model.

presented in [24]-[26]. We used the transient simulation method of TimeNET [24].

In our investigations, we used the same transition intensities for the existing transitions as in Section V.B, while the new transitions were set to the corresponding transition intensity values of the previous model. For example, transition *START_C* has the same intensity value as *START_A* and *START_B*.

The simulation results reflect 95% confidence level and 5% maximal error rating. Fig. 7 shows the cumulative expected value of the number of application purchases with regard to the different user types, while Fig. 8 depicts the cumulative expected value of the number of users who lost the interest in using the application as time elapses.

These results show similarity to the previous investigation, namely, approximately one third of the $Type_A$ users and 2% of the $Type_B$ users purchased the application. On the other hand, the $Type_C$ users' purchase depends on the number of free seeds in the network (and certainly on the initial parameter setting). According to our simulation results, 5% of the $Type_C$ users purchased the application, which ratio is much higher than the purchases of $Type_B$ users.

VI. CONCLUSION

In this paper, we presented two techniques for modeling application spreading aided by direct communication between the users' mobile devices.

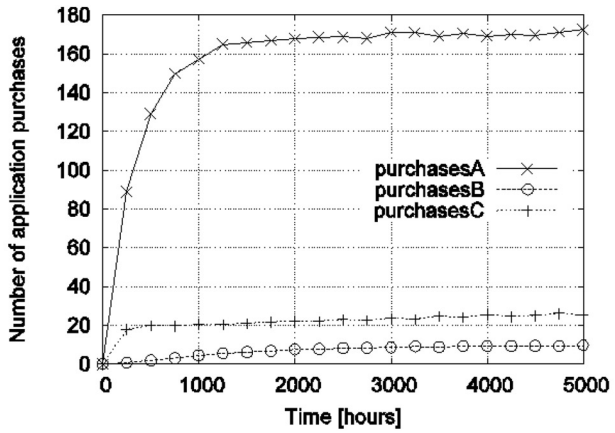

Fig. 7. Expected value of the number of application purchases as the function of elapsed time.

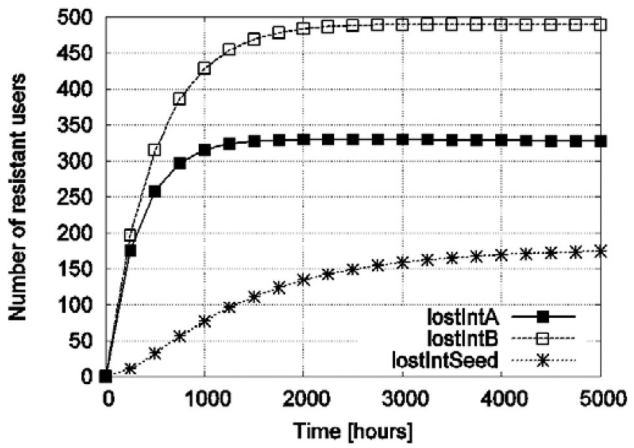

Fig. 8. Expected value of the number of users who lost the interest in using the application.

First, we presented our CQN model by which we can investigate the application spreading process assuming a homogeneous user behavior. This simple model allows a service provider to quickly calculate how much profit can be expectedly realized from application purchases.

Overcoming the limits of CQNs and being able to handle more complex user behaviors, we proposed two SPN models. In the basic one, we used the mean field based methodology to obtain an analytical approximation of the Petri net, which can derive results in the order of seconds. Then we presented an extended version of this basic Petri net that can accommodate an even more realistic user behavior for the price of using inhibitor arcs in the model. Unfortunately, the mean field approach cannot handle inhibitor arcs. Therefore, we investigated the extended Petri net model via simulations paying the fee of long runtime to produce results.

As we set the initial number of users from a given user type to the same (500) in every investigation, we can compare the results regarding to $Type_A$ and $Type_B$ users in the different models. The expected value of the number of $Type_A$ users' purchases was approximately the same in all models (169.97, 171.43 and 172.49), while this value of $Type_B$ users' purchases was also close to each other in the SPN models

(8.37 and 9.61). Therefore, we can consider the SPN models as extensions of the CQN model for cases when we have to handle more sophisticated user behaviors. On the other hand, we have to keep in mind that analytical results can be derived in a much faster way than producing results via running simulations, which can have also influence on selecting the model to be used.

In our models, with appropriate experience to set the model parameters a service provider can estimate his profit from application purchases in case of simple or even more complex scenarios. Moreover, he can learn the time behavior of the spreading process, by which he can realize additional gain, such as refining his marketing strategy.

Since the parameter setting is critical in this work, we plan to compare the outcome of our models to real data and refine the models accordingly as a future work.

REFERENCES

[1] T. G. Robertazzi, "Computer networks and systems: Queuing theory and performance evaluation," *New York: Springer-Verlag*, 1994.

[2] G. Balbo, G. Conte, S. Donatelli, and S. Franceschinis, "Modelling with generalized stochastic Petri nets," *John Wiley & Sons*, 1995.

[3] T. G. Kurtz "Solutions of ordinary differential equations as limits of pure jump Markov processes," *Journal of Applied Probability*, vol. 1, no. 7, pp. 49–58, 1970.

[4] Á. Horváth, and K. Farkas, "Modeling Self-Organized Application Spreading," *In Proc. of the 5th International ICST Conference on Access Networks (ACCESSNETS 2010)*, Budapest, Hungary, 2010.

[5] Á. Horváth, and K. Farkas, "Modeling Application Spreading using Mobile Ad Hoc Networks," *In Proc. of the Third Joint IFIP Wireless and Mobile Networking Conference (WMNC)*, Budapest, Hungary, 2010.

[6] M. Beccuti, M. De Pierro, A. Horváth, Á. Horváth, K. Farkas, "A Mean Field Based Methodology for Modeling Mobility in Ad Hoc Networks," *In Proc. of the 73rd IEEE Vehicular Technology Conference (VTC2011-Spring)*, Budapest, Hungary, 2011.

[7] Y. Wang, D. Chakrabarti, C. Wang, and C. Faloutsos, "Epidemic Spreading in Real Networks: An Eigenvalue Viewpoint," *22nd International Symposium on Reliable Distributed Systems (SRDS03)*, pp. 25-34, Florence, Italy, 2003.

[8] R. Pastor-Satorras, and A. Vespignani, "Epidemic Spreading in Scale-Free Networks," *Phys. Rev. Lett.*, 86:3200, 2001.

[9] F. Fu, L. Liu, and L. Wang, "Information Propagation in a Novel Hierarchical Network," *46th IEEE Conference on Decision and Control*, New Orleans, USA, 2007.

[10] A. Khelil, C. Becker, J. Tian, and K. Rothermel, "An Epidemic Model for Information Diffusion in MANETs," *5th ACM International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems*, Atlanta, Georgia, USA, 2002.

[11] O. Sekkas et al, "Probabilistic Information Dissemination for MANETs: the IPAC Approach," *20th Tyrrhenian Workshop on Digital Communications*, Pula, Italy, 2009.

[12] V. Karyotis, A. Kakalis, and S. Papavassiliou, "Malware-Propagative Mobile Ad Hoc Networks: Asymptotic Behavior Analysis," *Journal of Computer Science and Technology*, 23(3) pp. 389-399, 2008.

[13] E. Huang, W. Hu, J. Crowcroft, and I. Wassel, "Towards Commercial Mobile Ad Hoc Network Application: A Radio Dispatch System," *9th Annual International Conference on Mobile Computing and Networking*, San Diego, California, USA, 2003.

[14] F. Bai, and A. Helmy, "A Survey of Mobility Modeling and Analysis in Wireless Adhoc Networks," *Book Chapter in the book "Wireless Ad Hoc and Sensor Networks"*, Springer, October 2006, ISBN: 978-0-387-25483-8.

[15] S. Gowrishankar, T. G. Basavaraju, and S. K. Sarkar, "Effect of random mobility models pattern in mobile ad hoc networks," *International Journal of Computer Science and Network Security*, vol. 7, no. 6, pp. 160-164, 2007.

[16] M. K. J. Kumar, and R. S. Rajesh, "Performance Analysis of MANET Routing Protocols in different Mobility Models," *IJCSNS International Journal of Computer Science and Network Security*, p22, VOL. 9 No. 2, February 2009.

[17] B. Cohen, "Incentives Build Robustness in BitTorrent," *1st Workshop on Economics of Peer-to-Peer Systems*, UC Berkeley, California, USA, 2003.

[18] C. Bettstetter, "Mobility Modeling in Wireless Networks: Categorization, Smooth Movement, and Border Effects," *ACM Mobile Computing and Communications*, vol. 5, no. 3, pp. 55-67, July 2001.

[19] V. A. Karyotis, M. Grammatikou, and S. Papavassiliou, "A Closed Queueing Network Model for Malware Spreading over Non-Propagative Ad Hoc Networks," *In Proceedings of the 3rd ACM workshop on QoS and security for wireless and mobile networks (Q2SWinet '07)*, New York, USA, October 2007.

[20] M. Reiser, and S. S. Lavenberg, "Mean-Value Analysis of Closed Multichain Queuing Networks," *Journal of the ACM*, 27.2: 313-322, 1980.

[21] F. Bause, and P. S. Kritzinger, "Stochastic Petri Nets – An Introduction to the Theory," *Vieweg*, 2nd edition, 2002.

[22] G. G. Yin, and Q. Zhang, "Continuous-Time Markov Chains and Applications – A Singular Perturbation Approach," *New York: Springer-Verlag*, 1991.

[23] W. J. Stewart, "Introduction to the Numerical Solution of Markov Chains," *Princeton University Press*, 1995.

[24] A. Zimmermann, and M. Knol, "TimeNET 4.0: A Software Tool for the Performability Evaluation with Stochastic and Coloured Petri Nets: User Manual," *Technical Report, Technical University Berlin*, Real-Time Systems and Robotics Group, TR 2007-13, 2007.

[25] S. Baarir et al. "The GreatSPN tool: Recent Enhancements," *SIGMETRICS Perform. Eval. Rev.*, vol. 36, no. 4, pp. 4–9, 2009.

[26] P. Bonet, C. M. Llado, R. Puijaner, and W. J. Knottenbelt, "PIPE v2.5: A Petri Net Tool for Performance Modelling," *In Proceedings of the 23rd Latin American Conf. on Informatics (CLEI 2007)*, San Jose, Costa Rica, 2007.

Ádám Horváth received the M.Sc. degree in Computer Science from the Budapest University of Technology and Economics (BME) in 2007. After his M.Sc. he joined the Institute of Informatics and Economics at the University of West Hungary as a Ph.D. student under the supervision of Prof. Károly Farkas. His research interests cover security of wireless networks and mathematical modeling, mainly focused on Markovian processes.

Károly Farkas received his Ph.D. degree in Computer Science in 2007 from ETH Zurich, Switzerland, and his M.Sc. degree in Computer Science in 1998 from the Budapest University of Technology and Economics, Hungary. Currently he is working as an associate professor at Budapest University of Technology and Economics and at University of West Hungary, Sopron, Hungary. His research interests cover the field of communication networks, especially autonomic, self-organized, wireless and mobile ad hoc networks. He has published more than 50 scientific papers in different journals, conferences and workshops and he has given a plenty of regular and invited talks. In the years past, he supervised a number of student theses, participated in several research projects, coordinated the preparation of an EU IST research project proposal and acted as program committee member, reviewer and organizer of numerous scientific conferences. Dr. Farkas is a member of IEEE and fellow of the European Digital Media Academy (EADiM).