Proceedings of the Federated Conference on
Computer Science and Information Systems pp. 1239–1246

ISBN 978-83-60810-51-4

Simulating a Societal Information System for Healthcare

Hongying Du
University of South Carolina, 315
Main St. Columbia, SC 29208
USA
du5@email.sc.edu

Kuldar Taveter
Tallinn University of Technology,
Raja 15, 12618, Tallinn
Estonia
kuldar.taveter@ttu.ee

Michael N. Huhns
University of South Carolina, 315
Main St. Columbia, SC 29208
USA
huhns@sc.edu

***Abstract*—Societal information systems are intended to assist the members of a society in dealing with the complexities of their interactions with each other, especially regarding the resources they share. Because the members are distributed and autonomous, we believe that software agents, having these same characteristics, are a natural basis for representing the members and their interests in a societal information system. This paper describes a simulation of an agent-based societal information system for healthcare. Our design methodology is based on agent-oriented modeling, which we apply to analyze and design the system and its simulation. We execute the simulation to investigate four different strategies for assisting a person in choosing a physician, combined with three waiting strategies in three common social network models. The results show that the societal information system can decrease the number of annual sick days per person by 0.42-1.84 days compared with choosing a physician randomly.**

I. Introduction

THIS article concerns the simulation of a societal information system in the domain of healthcare. A societal information system is a large-scale information system that gathers information from hundreds or thousands of individual entities. Such systems can be abstracted as graphs with nodes representing individual entities and edges representing relationships between them. The purpose of a societal information system is to affect the behavior of a node by means of information retrieved from other nodes. Nowadays, a person's behavior is influenced by social networking services, such as Facebook. However, the amount of information to be comprehended and utilized in such services can be overwhelming for users. To further automate sharing and processing of information within a large social network or a societal information system, we are investigating supporting each node in the network by a software agent. Software agents are autonomous computational entities that can be viewed as perceiving their environment through sensors and acting upon their environment through effectors. To say that agents are computational entities simply means that they physically exist in the form of programs that run on computing devices. To say that they are autonomous entities means that to some extent they have control over their behavior and can act without the intervention of humans or other systems. Agents pursue goals or carry out tasks in order to meet their design objectives, and in general these goals and tasks can be supplementary as well as conflicting [1], [2]. Agents can form commitments and act on behalf of individuals and form multiagent systems (MAS). We view agent-based societal information systems as multiagent systems.

Societal information systems are appropriate for a wide variety of problems, including regulation (e.g., banking), allocation of scarce resources (e.g., electric power and parking spaces), distributed situation assessment (e.g., urban air quality), system control (e.g., traffic management, both vehicular and telecommunication), and decentralized decision-making (e.g., choosing medical care). This article addresses simulating a societal information system in the area of decentralized decision-making for healthcare.

Healthcare decision-making is done in many developed countries in the context of a healthcare quadruple, which consists of (1) patients, (2) healthcare providers (hospitals, health centers, labs, etc.) and provider networks, (3) insurance companies, and (4) the government. There is a variety of information systems available to support healthcare providers, provider networks, government healthcare agencies, and insurance companies, but none to support patients. Because patients are naturally distributed and are typically willing to assist each other, societal agent-based information systems instead of centralized information systems would be appropriate for fostering this mutual assistance. In such systems, each patient would be represented by a software agent. The agent would assist its principal in health-related activities, such as understanding and interpreting insurance rules, finding the most cost-effective insurer, finding a good healthcare provider, providing advice on cost-effective drugs and care, and monitoring the spread of disease symptoms and their treatments. Feedback and information sharing among patients would be used extensively in such systems.

Investigating societal information systems for healthcare is a broad research area. Moreover, it is difficult to experiment with such information systems in a society, especially because patients' health, privacy, and rights must be considered. We therefore have relied on simulations for prototyping and evaluating.

The second author expresses his gratitude to the Institute of International Education for supporting his work by a Fulbright Scholarship.

This article is organized as follows. First, we explain the method we use for prototyping the societal healthcare information system – agent-oriented modeling. Second, we describe briefly how agent-oriented modeling is applied to design a simulation of a societal information system for healthcare running on the NetLogo platform [3]. Third, we analyze and explain the simulation results. We conclude by comparing the outcomes of using different strategies in the healthcare system and discussing the benefits of a societal information system for healthcare.

II. Related work

Multiagent systems are widely used in different areas, such as tracking goods, traffic control, consensus knowledge, and decision-making [4]. One of the interesting areas for applying a MAS is healthcare.

Nealon and Moreno [5] analyze features of healthcare problems, including the distributed nature of the knowledge that is needed to solve a problem, coordination, complexity, and so on. They claim that a MAS is an appropriate approach to tackle healthcare problems and could be used for patient scheduling, organ and tissue transplant management, community care, information access, and decision support systems. Isern et al. [6] compare the internal architecture and communication-based coordination techniques of fifteen healthcare-related agent-based systems and claim that agent-based systems increase reusability, flexibility, and other beneficial qualities as compared with centralized software systems, such as client-server systems.

MASs are also broadly used in home-care systems. Koutkias et al. [7] present a MAS for monitoring and detecting important cases for disease management. Isern et al. [8] describe the K4Care Home Care model, which uses an agent-based platform. Grund et al. [9] describe a multiagent monitoring system that uses data mining techniques to determine the status of a patient. Charfeddine [10] introduces an agent-oriented framework to simulate the population of a chronic disease.

In the work most closely related to ours, Udupi and Singh [11] use conceptual models in a societal information system to implement a peer-to-peer network in which an agent contacts other agents to discover suitable service providers. It uses InterPol, a language and framework for supporting different kinds of interaction policies between agents. In [12], we described the modeling method of our societal information system.

There are several websites, similar to RateMDs [13], where people rate doctors according to punctuality, medical knowledge, and other characteristics, and add comments. As we explain later, our approach differs from such websites and has advantages.

III. Methodology

This article focuses on designing societal information systems of a particular kind – societal information systems for finding an appropriate physician and finding out the benefits to do so. We use the case study method [12] and explore by rapid prototyping the design of a simulation of a societal information system for healthcare. Rapid prototyping stands for implementing a proof-of-concept prototype in an agile way by directly mapping the modeling constructs to the constructs of a scripting environment like Netlogo or some agent-oriented environment like JADE. The method we use for prototyping is agent-oriented modeling. Agent-oriented modeling as described in [14] is a holistic approach for analyzing, designing, and rapid prototyping of societal information systems consisting of humans and technical components. We have chosen agent-oriented modeling because it is geared towards prototyping distributed systems that are open, adaptive, and intelligent. Societal information systems are *open* systems because members of the society (e.g., commuters, patients, or shoppers) may join and leave the system at any time. Societal information systems are *adaptive* systems, because they should react to their constantly changing environment, which for example can take the form of changes in traffic infrastructure, health insurance coverage, and product prices. We also term societal information systems as *intelligent* systems, because they reflect the "wisdom of crowds" when recommending a patient, for example, a healthcare provider. In addition, agent-oriented modeling meets well the requirements for purposefulness and understandability of the design.

A set of canonical models are introduced in agent-oriented modeling, whose types are shown in Table I. In addition to representing each model with an abstraction layer (analysis, design, or prototyping), Table I maps each model to the vertical viewpoint aspect of interaction, information, or behavior. Each cell in the table represents a specific viewpoint. We explain these viewpoints in the following paragraphs.

From the viewpoint of *interaction analysis*, *role models* represent the properties of roles and the relationships between the roles are represented by an *organization model*. From the viewpoint of *information analysis*, a *domain model* represents the knowledge to be handled by the societal information system. From the viewpoint of *behavior analysis*, a *goal model* is a container of three components: goals, quality goals, and roles.

From the viewpoint of *interaction design*, *agent models* transform the abstract constructs from the analysis stage, roles, to design constructs, agent types, which will be realized in the implementation process. *Interaction models* are used to express interaction patterns between agents. From the viewpoint of *information design*, *knowledge models* represent both private and shared knowledge of agents. From the viewpoint of *behavior design*, *behavioral*

scenarios are used to show how agents make decisions and perform activities [15].

Modeling at the abstraction layer of prototyping is explained in Section IV.

TABLE I.
THE MODEL TYPES OF AGENT-ORIENTED MODELING

	Viewpoint aspect		
Abstraction layer	Interaction	Information	Behavior
Analysis	Role models and organization model	Domain model	Goal models
Design	Agent models and interaction models	Knowledge models	Behavioral scenarios
Prototyping	Interaction prototyping	Information prototyping	Behavior prototyping

Fig. 1 shows the goal model of our societal information system for healthcare, in which rectangles stand for functional goals and clouds stand for quality goals. The stick figures represent roles that are required for achieving the goals. As can be seen from Fig. 1, from the viewpoint of *behavior analysis*, our societal healthcare information system focuses on the purpose of "Allocate Healthcare Resources" among the members of the society. Specifically, we study the allocation of physicians – a special kind of healthcare resource. Achieving the functional goal "Allocate Healthcare Resources" is characterized by the quality goal "Maximal Societal Health", which determines the quality criterion according to which healthcare resources should be allocated in a society.

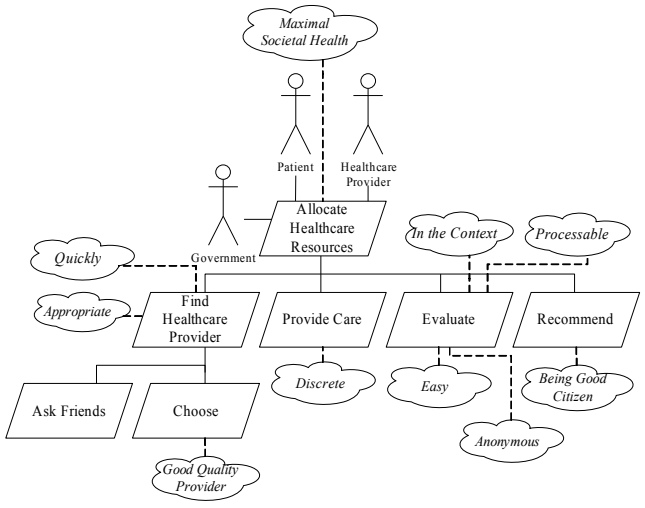

Fig. 1 The goal model of the societal information system

To accomplish the purpose "Allocate Healthcare Resources" of the societal information system, its four subgoals need to be achieved: finding a healthcare provider, being provided with care, evaluating the care, and recommending healthcare providers to other patients. As we demonstrate below, to fulfill the goal "Find Healthcare Provider", a patient recursively asks her friends, friends' friends, and so forth for recommendations and chooses the best physician recommended. This is represented as two subgoals of "Find Healthcare Provider:" "Ask Friends" and "Choose."

We attach a number of quality goals to the functional goals in the goal model. The meanings of the quality goals are easy to understand. For example, "Quickly" means a patient wants to find a healthcare provider as soon as possible. The "Anonymous" quality goal expresses that no evaluation by a patient should identify the patient. It should be noted that the quality goal "In the Context" attached to the functional goal "Evaluate" represents that evaluation has to occur in the context of receiving the service, preferably before leaving the facilities of the healthcare provider or at least on the same day. The "Processable" quality goal means that the evaluation should be presented in a form amenable to computer processing. In our simulation, we use a scale from 1 to 5 to measure the evaluations.

According to Fig. 1, we model two roles for our simulation – Patient and Healthcare Provider. There is also a third role – Government. Since our work focuses on the particular aspect of the U.S. healthcare domain dealing with how a patient finds a physician, rather than modeling the healthcare domain in its full complexity, the Government role's modeling is not relevant to the simulation system being designed and we ignore the Government role in our system. Additionally, we complement the goal model with the new Assistant role, which is not shown in Fig. 1. The Assistant role is the assistant of a person and is responsible for asking friends for recommendations, choosing a healthcare provider, and assisting in evaluating the care. In the prototypical system being designed, the role of Assistant should obviously be mapped to the Assistant Agent software agent type. Since a patient is a real human that is treated by another real human – a physician – we map both the roles Patient and Healthcare Provider to the Human Agent type. The software system boundary of the societal information system is obviously between the roles Patient and Assistant.

From the viewpoint of *interaction analysis*, the organization model of the societal information system being designed is decided based on the three kinds of networks that are used for representing the relationships among the members of the society:

- Random network: the relationships between pairs of patients are created randomly.

- Small-world network: most nodes are not neighbors to one another, but most nodes can be reached from any other node by a small number of hops [16].

- Scale-free network: the shortest paths between nodes flow through hubs, and if a peripheral node is deleted, it is unlikely that this will interfere with passing a message between other peripheral nodes.

We use the Barabási–Albert model [17] to construct a scale-free network for our simulation. A scale-free network is a common model for a collaboration network.

After covering the viewpoints of behavior analysis and interaction analysis, we next proceed to the viewpoint of *information analysis* by addressing the knowledge to be represented within the system. We do this by identifying the types of knowledge entities related to the roles. As each healthcare provider has predefined capacity and efficiency, which are explained in Section IV, we attach the Capacity and Efficiency knowledge entity types to the Healthcare Provider role.

We now proceed to the viewpoint of *interaction design*. Finding a physician involves interactions between Assistant Agents representing patients. We represent these interactions as an interaction protocol between agents of the type Assistant Agent. It is appropriate to remind here that the difference between an interaction protocol and other kinds of interaction models is that an interaction protocol models some aspects of the agent behaviors along with their interactions [14].

Representing the interaction protocol of the societal healthcare information system is very important, because it describes the patient's strategy of choosing a physician. We explored the following four possible strategies:

- Random strategy. The patient's Assistant Agent randomly chooses a physician.

- The "Choose one" strategy. The patient's Assistant Agent chooses the best physician according to the patient's evaluations for physicians. If the patient has no evaluations, his/her Assistant Agent asks his/her friends' Assistant Agents for recommendations. The Assistant Agent acting on behalf of the patient's friend may deal with the request in one of the following ways:

o Reply with a recommendation.

o Provide the requesting agent with the address of the Assistant Agent of one of its principal's friends if there is no recommendation to give. This process continues recursively until the first recommendation is received or until all the friends down to the maximum forwarding depth have been asked. The forwarding depth is defined as follows: the originator's friends are at depth 1; the originator's friends' friends at depth 2, and so on.

Fig. 2 presents the interaction protocol among patients' Assistant Agents for the "Choose one" strategy. It models that the Assistant Agent of a patient's friend may respond with a recommendation or recommend the Assistant Agent of the friend's friend. This means the interaction protocol is recursive, which is represented by the "Loop" behavioral construct. A friend's Assistant Agent may also ignore a request, which is not shown in the figure.

In addition to the random and "Choose one" strategies, we have included in our simulations the "Borda voting" and "Add and minimize" strategies. These strategies are briefly described as follows:

- The "Borda voting" strategy. The patient's Assistant Agent asks his/her friends' Assistant Agents, who are closer than a specified limit, for recommendations. A friend's Assistant Agent may choose to answer or refrain from answering just like with the "Choose one" strategy. After the patient's Assistant Agent has received all the responses, it calculates for each physician the Borda count [a single-winner election method in which voters rank candidates in order of preference, named for the 18th-century French mathematician and political scientist Jean-Charles de Borda, who devised the system in 1770], according to which a physician is given a number of points equal to the number of physicians whose evaluations are worse than the evaluations of the given physician. Thereafter the agent adds up all the points gained by the physician in question. The physician with the highest score is chosen.

- The "Add and minimize" strategy, which has the same procedure for getting recommendations as the "Borda voting" strategy. After the patient's Assistant Agent has received all the responses, it adds up all the non-zero evaluations and calculates the mean value of them for each physician. Then the Assistant Agent chooses the physician with the minimum mean evaluation. Choosing the physician with the minimum value is due to the way we define the evaluation, as described in Section IV.

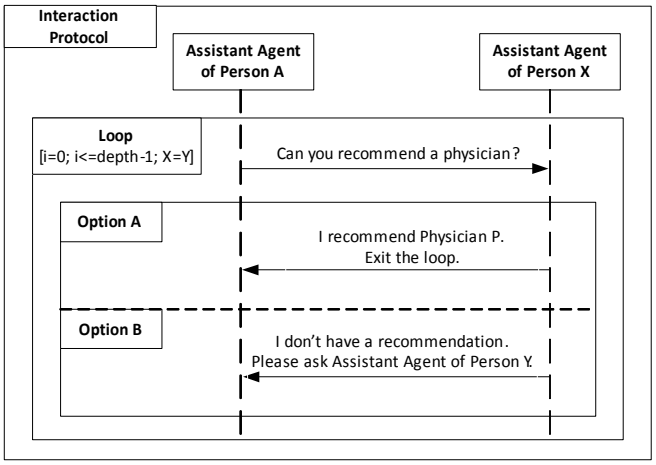

Fig. 2 The interaction protocol for "Choose one" strategy

From the viewpoint of *behavior design*, to model the behaviors of agents of the decided types, we transform responsibilities of the roles into activities attached to the agent types. As a result, we obtain behavioral scenarios for agents playing the roles Patient, Assistant, and Physician.

For example, the behavioral scenario of an agent of the type Assistant Agent playing the role Assistant models that the activities "Find a physician" and "Evaluate" are performed sequentially. In societal information system for healthcare this is always the case, because the Assistant Agent does not perform any activities between these activities while a patient is attended by a physician.

Another aspect of the Assistant Agent's behavior in choosing a physician deals with what the agent should do if the physician is not available on the given day. We have decided to consider the following three waiting strategies of a patient:

- Waiting. The patient's Assistant Agent chooses the best physician by adopting one of the strategies of choosing a physician explained above and sticks to this choice. If the physician is busy, the patient will still make an appointment with the physician and will wait until the physician becomes available.

- No waiting. If the physician chosen is busy, the patient's Assistant Agent will choose a physician randomly according to the "Random" strategy or the next best physician according to the other physician-choosing strategies until it finds an available physician.

- Waiting with limit. If the physician chosen is not available, the patient's Assistant Agent will check whether the physician could be reached in a certain number of days. If it is possible, the patient will make an appointment and wait. If not, the Assistant Agent will choose another physician according to the rules of the same waiting strategy. If no physician is available in a certain number of days, the Assistant Agent will choose a physician who has the smallest number of days required for waiting.

Finally, distinguishing between private and public knowledge entities from the viewpoint of *information design* is straightforward, because the knowledge entity Evaluation is private to the patient and Assistant Agent helping him/her, while the knowledge entity Recommendation is shared between different patients and instances of Assistant Agent. Similarly, the knowledge entity Efficiency is private to each Healthcare Provider, but at the same time naturally forms a basis for how patients evaluate healthcare providers.

We describe models including role models, the organization model, and the domain model in detail in [12].

IV. Evaluation

A. Simulation Settings

We next describe from the three viewpoints introduced in Section III how we mapped agent-oriented models of the societal information system to the programming constructs of the simulation environment – NetLogo.

From the viewpoint of *information prototyping*, we represented the knowledge entities decided by agent-oriented modeling as described in Section III as follows:

- The Capacity knowledge entity – in terms of the number of patients per day that a given physician can handle.

- The Efficiency knowledge entity – in terms of the number of days that it takes for a given physician to cure a patient. This number of days is generated for each physician according to a normal distribution whose mean and standard deviation can be adjusted in the user interface.

- The Evaluation knowledge entity – in terms of the following variables:

o The number of days the physician in question failed to handle a given patient. How this value is determined is explained below.

o The number of days that the physician needed to cure a patient. This is determined by the Efficiency knowledge item pertaining to the physician.

o A random component representing that different patients evaluate the same physician differently.

A patient's evaluation for a specific physician is calculated by adding these three factors. For example, let us assume that a patient gets sick today and decides to visit a physician chosen by her Assistant Agent, but the physician is busy and cannot see the patient until tomorrow. In this case, the value of the first factor – the number of days the physician in question failed to handle a given patient – is 1, because the patient had to wait for 1 day to see the physician. The second factor – the number of days that the physician requires to cure the patient – is a fixed number related to the physician in question. The third factor – the random component expressing the subjective factor – is a random value that varies between -0.5 and 0.5.

The viewpoint of *behavior prototyping* covers the behaviors of software agents representing patients and physicians. In accordance with the behavioral scenarios modeled as a part of the design described in Section III, every day the patients each try to decide which physician to visit. For each patient, the Assistant Agent acting on behalf of its principal may ask Assistant Agents of the principal's friends for recommendations and then makes a decision as to which physician the principal should visit.

From the viewpoint of *interaction prototyping*, the exchange of messages to be implemented is modeled according to interaction diagrams, such as the one in Fig. 2 for the case of choosing a physician according to the "Choose one" strategy. To make our simulations more realistic, we have chosen a 20% probability that a friend would ignore the patient's request.

Back to the viewpoint of *behavior prototyping*, the software agent corresponding to the Assistant Agent recommends physicians based on evaluations. The agent can recommend only those physicians that its principal has actually visited in the simulation. The number of days the physician in question could not handle the given patient, because of the physician's exceeded capacity, accumulates in the patient's evaluation until the patient actually visits the given physician. On each new visit the agent "forgets" its

previous evaluation and updates its knowledge base with the new evaluation. The reason why the agent forgets its previous evaluation is that during the time period between the previous evaluation and the new evaluation, factors that influence the evaluation may have occurred. For example, the physician may have become more skilled. Therefore it is fairer to use the latest evaluation.

To make our simulations as realistic as possible, we used the following statistical data by the Centers for Disease Control and Prevention (CDC) from the year 2008 [18]:

- The number of physician office visits per 100 people per year: 320.1.
- The number of physicians per 10,000 people: 26.

Based on the above data, we obtained the average number of people who get sick every day by dividing the number of visits per 10,000 people by 250, which is the standard number of working days in a calendar year in the U.S. As a result, 128 people out of a population of 10,000 get sick every day.

B. Results and Evaluation

We simulated 365 days with 5,000 patients and 13 physicians. In our simulation, 64 random people get sick every day. The value of the local variable of each physician's software agent corresponding to the Capacity knowledge entity was set to 8 patients per day. The value of the local variable of each physician's software agent corresponding to the Efficiency knowledge entity was determined randomly according to a normal distribution with mean value 3 days and with the value of deviation as 2.0.

Fig. 3 describes the number of days needed for curing by different physicians in our simulations.

We performed simulations by combining the types of social networks explained in Section III with different strategies of choosing a physician and waiting strategies, which are both described in Section III. The results from simulations in terms of the annual sick days per person and leftover patients who were not taken care of by the end of the last day simulated are represented in Tables II - VII.

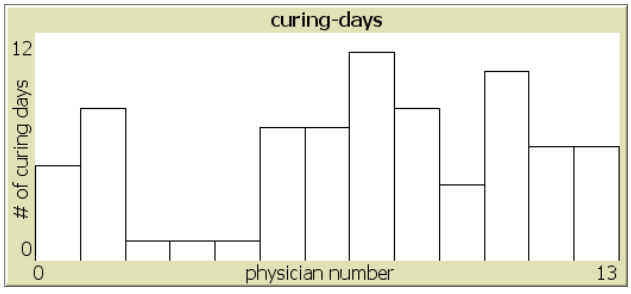

Fig. 3 Days needed to be cured by different physicians

TABLE II.
AVERAGE SICK DAYS FOR RANDOM NETWORK

	Waiting	No-waiting	Waiting-with-limit
Random	6.8	6.78	6.80
Choose one	12.18	5.21	6.34
Borda voting	34.75	6.01	8.31
Add and minimize	10.17	4.94	6.37

TABLE III.
LEFTOVER PATIENTS FOR RANDOM NETWORK

	Waiting	No-waiting	Waiting-with-limit
Random	0	0	0
Choose one	427	0	80
Borda voting	4569	0	96
Add and minimize	155	0	94

TABLE IV.
AVERAGE SICK DAYS FOR SMALL-WORLD NETWORK

	Waiting	No-waiting	Waiting-with-limit
Random	6.79	6.78	6.79
Choose one	10.31	5.22	6.31
Borda voting	7.63	6.58	6.83
Add and minimize	9.59	4.96	6.34

TABLE V.
LEFTOVER PATIENTS FOR SMALL-WORLD NETWORK

	Waiting	No-waiting	Waiting-with-limit
Random	0	0	0
Choose one	407	0	71
Borda voting	76	0	25
Add and minimize	239	0	93

TABLE VI.
AVERAGE SICK DAYS FOR SCALE-FREE NETWORK

	Waiting	No-waiting	Waiting-with-limit
Random	6.79	6.74	6.77
Choose one	13.72	5.29	6.33
Borda voting	38.56	6.49	9.92
Add and minimize	9.41	4.92	6.35

TABLE VII.
LEFTOVER PATIENTS FOR SCALE-FREE NETWORK

	Waiting	No-waiting	Waiting-with-limit
Random	1	0	2
Choose one	686	0	78
Borda voting	4449	0	110
Add and minimize	255	0	96

We can see from Tables II - VII that if a patient adopts the "Waiting" strategy, the "Random" strategy will outperform all the other strategies of choosing a physician in social networks of all three kinds addressed. This is the case because in all the other strategies, a patient always waits for the best physician chosen by her Assistant Agent, which increases the waiting days and accordingly sick days. Also, because the Random strategy leads to even visiting of physicians, it has no leftover patients, but there are leftover patients in the other three strategies.

The performance of the "Borda voting" strategy is the worst in all three kinds of social networks addressed, except for the combination of the "Borda voting" strategy and "Waiting" strategy in the small world network, because it uses more evaluation information than the other strategies due to its method for calculating the votes for physicians.

Differently from the "Borda voting" strategy, the "Add and minimize" strategy uses less information because it does not consider the physicians who have not been evaluated. The patients whose Assistant Agents follow the "Add and minimize" strategy therefore tend to choose physicians with fewer days required for curing, as compared with other physician choosing strategies, and then wait for that physician chosen, which increases the number of sick days.

If the "No waiting" strategy is adopted, all the other strategies will outperform the "Random" strategy. This is because the patients' Assistant Agents consider the evaluations by their principals' friends and choose the best physicians, and there is no problem of waiting.

If the "Waiting with limit" strategy is adopted, the "Choose one" and "Add and minimize" strategies of choosing a physician show the best performance. However, these strategies result in more leftover patients than the random strategy. This is reasonable, since according to "Choose one" and "Add and minimize" strategies a patient may be willing to wait for a good physician if the waiting time is less than 2 days, leading to just a few leftover patients and less average annual sick days.

According to the "Random" strategy of choosing a physician, patients' Assistant Agents just choose physicians randomly and each physician has almost the same number of patients in total. For the other three strategies, as time passes, Assistant agents gradually gather enough information about physicians, evaluate them, and recommend to their friends the best physicians they are aware of. As a result, after patients have formed their opinions about the physicians, good physicians get full capacity of patients every day and bad physicians get only a few patients. Due to space limitations, the graphs showing this trend are not shown here.

We also performed simulations with fewer physicians to check whether the claims stated above still hold. We adopted the "Choose one" and "Add and minimize" strategies of choosing a physician and "No waiting" and "Waiting with limit" strategies for conducting simulation experiments with 7 physicians. Table VIII shows the results in terms of average sick days. We can see that for the "No waiting" strategy, the "Choose one" and "Add and minimize" strategies of choosing a physician still perform better than the combination of "Random" and "No waiting" strategies. This is because a patient's Assistant Agent first chooses a physician who requires less days for curing and only then randomly chooses a physician if the patient has to wait.

Table VIII also shows that the combination of the "Waiting with limit" strategy and all strategies for choosing a physician yields almost the same result.

TABLE VIII.
AVERAGE SICK DAYS WITH SEVEN PHYSICIANS

	No-waiting	**Waiting-with-limit**
Random	28.01	19.23
Chose one	26.15	19.24
Add and minimize	18.33	19.24

In addition, we investigated the performance of a system having a lower probability that the friends of a patient in small world network will answer a request, which are shown in the following two tables. The "Random" strategy is shown here because it is not influenced by the probability.

TABLE IX.
AVERAGE SICK DAYS WITH PROBABILITY = 0.6

	Waiting	**No-waiting**	**Waiting-with-limit**
Choose one	9.86	5.31	6.29
Borda voting	7.60	6.26	6.60
Add and minimize	9.75	5.01	6.37

TABLE X.
AVERAGE SICK DAYS WITH PROBABILITY = 0.4

	Waiting	**No-waiting**	**Waiting-with-limit**
Choose one	8.72	5.42	6.24
Borda voting	7.05	6.07	6.42
Add and minimize	10.38	5.14	6.40

Comparing Table IV with these two tables, we discovered that the conclusions before still hold. To clarify this, we include Table XI, which denotes the changing trend of the average sick days while the probability is decreasing. In the table, "+" means increasing and "-" means decreasing. We can see from the table that for "Add and minimize", the average sick days increases while the probability decreases, because patients gets fewer responses from friends, leading to less-informed decisions. For the "Borda voting" strategy, the average sick days decreases while the probability decreases. As mentioned before, due to the way that the "Borda voting" strategy gets the evaluation and calculates, it

always gets too much information and lead to worse results than other strategies. So when the information is less with decreasing probability, we have better results of less average sick days. There's no fixed trend for a certain waiting strategy with different physician choosing strategies.

TABLE XI.
CHANGING TREND WITH DECREASING PROBABILITY

	Waiting	**No-waiting**	**Waiting-with-limit**
Choose one	-	+	-
Borda voting	-	-	-
Add and minimize	+	+	+

V. CONCLUSIONS

This paper describes the design and rapid prototyping of a societal information system for healthcare. Agent-oriented modeling was chosen for developing our simulation because it explicitly addresses the design of societal information systems where the activities of humans are supported by software agents.

We investigated the prototyped societal healthcare information system using agent-based simulations on the NetLogo platform. The NetLogo code for the simulation can be found in [19]. In the simulation, we investigated the influence of different strategies of finding an appropriate physician and different waiting strategies in three common social network models. Our prototype revealed that if a patient adopts the "Waiting" strategy, the "Random" strategy will outperform all the other strategies of choosing a physician. On the other hand, with the "No waiting" strategy, all the other strategies will outperform the "Random" strategy. If the "Waiting with limit" strategy is adopted, the "Choose one" and "Add and minimize" strategies will show the best performance. We found that by adopting the "Choose one" or "Add and minimize" strategy, in the "No-waiting" and "Waiting-with-limit" case, the average number of sick days can be reduced by 0.42-1.84 days or 6.2%-27.1%.

Our societal information system differs from RateMDs and other similar websites where people can rate and find physicians in the way that people rate the physicians and patients interact. It is difficult to compare the effect of these websites and that of our system, mostly because there are no objective evaluation statistics on the websites, such as the length of time each patient takes to get cured during a period. Such websites use more flexible criteria on which different people might have different opinions, such as punctuality, medical knowledge, and time spent on a patient, while our system uses the time it takes to cure a patient as a criterion, which is more objective and meaningful. Although patients might access more ratings online, they usually do not know the people who have rated the physicians and there is a higher possibility that the ratings will not be truthful and accurate. In our system, a patient relies on friends' recommendations, which are typically more reliable.

ACKNOWLEDGEMENTS

We express our gratitude to Professor Leon Sterling from Swinburne University of Technology, Australia for comments and suggestions on a draft of this paper.

REFERENCES

[1] M. N. Huhns, and L. M. Stephens, (1999). Multiagent systems and societies of agents. In *G. Weiss (ed.), Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Cambridge, MA, and London, England: MIT Press.

[2] M. Wooldridge, (2009). *An Introduction to Multiagent Systems*. 2nd Edition. Chichester, UK: John Wiley & Sons.

[3] U. Wilensky, (1999). *NetLogo*. Last accessed on February 16, 2012, from http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling. Evanston, IL: Northwestern University.

[4] M. N. Huhns, (2009). From DPS to MAS to ...: continuing the trends. In *C. Sierra, C. Castelfranchi, K. S. Decker, & J. S. Sichman (eds.), 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May 10-15, Volume 1 (pp. 43-48)*. New York, NY: ACM.

[5] J. Nealon and A. Moreno. "Agent-Based Applications in Healthcare", in *Applications of Software Agent Technology in the Healthcare Domain*, chapter 2, pp. 3–18. Birkhäuser Verlag, 2003.

[6] Isern, D. Sánchez, A. Moreno, "Agents applied in healthcare: a review." *Int. J. Med. Inf*. vol. 79, no. 3, 145–166, Mar. 2010.

[7] G. Koutkias, I. Chouvarda, N. Maglaveras, "A multiagent system enhancing home-care health services for chronic disease management," *IEEE Trans. Information Technology in Biomedicine*, vol. 9, no. 4, pp. 528–537, Dec. 2005

[8] Isern, A. Moreno, G. Pedone, L. Varga, "An intelligent platform to provide home care services," in *Proc. of the 2007 conference on Knowledge management for healthcare procedures*, July 07, 2007. Lecture Notes on Computer Science, vol. 4924, pp. 149-160, Amsterdam, The Netherlands.

[9] M. Gund, S. Andhalkar, D. Patil, V. M. Wadhai, "An intelligent architecture for multi-agent based m-healthcare system," *Int. Journal of Computer Trends and Technology*, vol. 1, no. 1, 2011.

[10] M. Charfeddine, B. Montreuil, "Integrated agent-oriented modeling and simulation of population and healthcare delivery network: application to COPD chronic disease in a Canadian region." in *Proc. Of the 2010 Winter Simulation Conf.*, Quebec, Canada, pp. 2327–2339, Dec. 2010.

[11] Y.B. Udupi and M.P. Singh, "Information Sharing among Autonomous Agents in Referral Networks," in *Agents and Peer-to-Peer Computing, 6th International Workshop (AP2PC 2007)*, S.R. Joseph, Z. Despotovic, G. Moro and S. Bergamaschi, eds, Lecture Notes in Computer Science, Vol. 5319, Springer-Verlag, 2007, pp. 13–26.

[12] K. Taveter, H. Du, and M. N. Huhns. "Method for rapid prototyping of societal information systems." In *Proceedings of Workshop on Agent Based Computing: from Model to Implementation IX (ABC:MI),* Wrocław, Poland, September 9-12, 2012.

[13] *RateMDs*. (2012). Find and Rate Doctors and Dentists, http://www.ratemds.com/, retrieved: 15th January 2012.

[14] L. Sterling, and K. Taveter, (2009). *The Art of Agent-Oriented Modeling*. Cambridge, MA, and London, England: MIT Press.

[15] K. Taveter, H. Du, and M. N. Huhns, "Engineering societal information systems by agent oriented modeling," *J. of Ambient Intelligence and Smart Environments*, vol. 4, no. 3, pp. 227-252, 2012.

[16] J. Watts, and S. H. Strogatz (1998). Collective dynamics of 'small-world' networks. *Nature, 393*, pp. 440-442.

[17] A.-L. Barabasi and R. Albert (1999). Emergence of Scaling in Random Networks. *Science, 286(5439)*, pp. 509–512.

[18] *Ambulatory Care and Physician Visits*. (2011). Last accessed on June 19, 2011, from http://www.cdc.gov/nchs/fastats/docvisit.htm.

[19] http://www.cse.sc.edu/~huhns/HealthCarePhysicianV9full.nlogo