BULGARIAN ACADEMY OF SCIENCES
PROBLEMS OF ENGINEERING CYBERNETICS AND ROBOTICS • 2024 • Vol. **82**, pp. 3-20
p-ISSN: 2738-7356; e-ISSN: 2738-7364
https://doi.org/10.7546/PECR.82.24.01

Information System for Generating Schedules for Preventive Examinations: An Algorithmic Implementation

Vasil Dimitrov[1], Zornitsa Dimitrova[1]

[1] Institute of Information and Communication Technologies at the Bulgarian Academy of Sciences, Sofia-1113, Bulgaria
Emails: vasil.zdimitrov@gmail.com, zornitsa.dimitrova@iict.bas.bg

Abstract: One of the key priorities of the national health strategy is related to strengthening the capacity of public health, as the main policy of this priority is prevention. Prevention is an important activity that reduces the risks to public health by preventing a disease or, if it occurs, preventing complications. In this regard, the problem of generating a schedule for annual preventive medical examinations by a general practitioner (GP) is particularly important. Therefore, this article elicitates the requirements and describes a specific task for developing an information system for generating a schedule for preventive examinations. The described algorithms have been tested in the developed demo version to verify their applicability and operability. The obtained results are encouraging, and therefore the algorithmic implementations could be successfully used to construct a unified system for generating schedules for preventive medical examinations.

Keywords: Schedule generation, preventive examinations, software system, requirements elicitation

1. Introduction

With the development of information and communication technologies, more and more activities rely on the use of various smart technologies [1, 2]. On the other hand, more and more information systems are migrating to the cloud, which is a prerequisite for creating new business models [3, 4]. The SaaS business model [5] allows for quick and easy distribution of information systems without the need to organize a distribution network, maintenance, etc. A relatively new area should

also be noted here – smart healthcare, which uses technologies such as IoT, big data, AI and ML to improve the quality and efficiency of healthcare delivery [6]. All this leads to an increase in the variety of information systems capable of collecting, organizing, processing and storing information [7, 8, 9].

The standards and requirements of users for working in an online environment are constantly evolving, and with them the technologies for the development of web applications. Changes are most visible in the user interface, but they are only possible thanks to the development of technology and the transformation of web architectures [10]. The development of web applications is not a simple and fast process, it is more of an evolution than a revolution. New app architectures don't make major changes to existing ones, but instead gradually improve features over time.

Why are information systems important? This is due to several reasons, which can be summarized as follows:

1. *Improving Decision-Making:* They provide accurate and timely information that helps managers make informed decisions.
2. *Efficiency and Productivity:* Automate routine tasks, which saves time and resources, allowing employees to focus on more important activities.
3. *Data Management:* Systems organize, store, and analyze large amounts of data, making it easier to manage and access information.
4. *Improve communication:* Information systems facilitate the exchange of information between employees and departments, which improves teamwork.
5. *Analysis and Forecasts:* They allow for the performance of analyzes and forecasts, which is important for strategic business planning.
6. *Competitive Advantage:* Provide opportunities to innovate and adapt to changes in the market, which can provide a competitive advantage.
7. *Risk Management:* Help identify and manage business-related risks.

Information systems are key to the successful functioning of modern organizations, including in the healthcare sector [11]. Information systems help to track patients' medical records, which facilitates access to important information and improves coordination between different professionals [12]. The creation of electronic health records allows for faster and more efficient information management while reducing the risk of data loss. The systems can perform analyses of large volumes of medical data [13], which supports research and discoveries related to new treatments and diseases. Information systems allow healthcare professionals to monitor and evaluate the quality of services provided, resulting in better outcomes for patients. Information systems facilitate remote treatment and consultation through fast and efficient communication between healthcare professionals and patients, which increases the quality of care [14],

which is particularly useful in remote areas or during emergencies. Considering the sensitivity of the data to be worked with, it is necessary to ensure appropriate information security when designing any system [15, 16].

Information systems are the basis for optimizing the management of hospitals and healthcare facilities, including inventory management, staff schedules and budgeting. Considering the importance of preventive examinations and the workload of general practitioners, this article describes an algorithmic implementation of an information system for generating schedules for preventive examinations.

2. Specific requirements for information systems for generating a schedule for preventive examinations

The essence of information systems is closely related to data structures [17]. Therefore, it is important to define in advance the requirements for the information system for generating a schedule for preventive examinations. These requirements can be divided into three main groups: functional, non-functional and requirements imposed by the environment.

2.1. Functional requirements

The functional requirements reflect the different actions that the application must perform. This group includes the main purpose of the application – generating a schedule, filling in and editing the database, managing the processes of notifying patients about their scheduled hours, as well as other support operations. The functional requirements elicitation for the information system for generating a schedule for preventive examinations are as follows:

- User registration;
- Adding a patient with all attributes;
- Adding a patient only with mandatory attributes;
- Adding patients from .xlsx file;
- Editing an existing patient;
- Deleting an existing patient;
- Search for a patient by name;
- Generation of a schedule for examinations for a calendar year;
- Visualization of a generated schedule for: Month, Week, Day;
- Notification of patients by e-mail;
- Assigning a text to an e-mail in addition to the automatically generated one;
- Notification of patients via SMS;

- Assign SMS text in addition to the automatically generated one;
- Marking a patient as having undergone an examination;
- Postpone examination for automatically selected new date and time.

2.2. Non-functional requirements

The information system for generating a schedule for preventive examinations works with personal data falling within the scope of legal regulations, various rules and requirements for access and protection of information must be observed. The non-functional requirements that the proposed information system must have are:

- Support for a password-protected doctor's account;
- Connection encryption;
- Possibility for simultaneous operation of more than one user;
- Ability to access the application through comp network;
- Ability to import an existing database, which is done by a system integrator, not a user.

2.3. Requirements imposed by the environment

These are a group of requirements that are gaining more and more weight as the functionality is enriched and the scale of the system grows and must be considered when choosing technologies and architecture. The requirements imposed by the environment are related to the following features:

- The goal of the system is to save time by eliminating the need for a person to create a schedule "by hand", so all operations should be designed in such a way that they require minimal interaction with the user;
- The users of the system will generally be without IT knowledge, which imposes requirements for a simple and intuitive user interface. Each operation must be decomposed into short and clear steps. All input data must go through strict validations.

3. General concept of an information system for generating a schedule for preventive examinations

3.1. Roles

The only role in using an information system to generate a schedule for preventive examinations is that of the user. This can be a general practitioner, nurse or administrative staff member who is responsible for generating a schedule for preventive examinations and monitoring its implementation. To run any of the

workflows, the same access rights are required, making it unnecessary to introduce other roles.

It is necessary to have an information system that has such a design and architecture that allows the creation of new roles with different rights, without affecting the existing functionality or the need to stop working. A new type of user can be, for example, patients to check their scheduled appointment and appointment online, but for the initial version this role is not introduced.

3.2. Workflow processes

The workflows in the information system for generating a schedule for preventive examinations are the individual operations that the user must perform sequentially to obtain a certain result. All processes are executed by the user's role, as it is the only one. For some of the operations, it is mandatory that preconditions have been met before they are started, which make their execution possible.

- Creating a user account – this is the first process that must be completed to be able to perform all other activities. The result of this process is the creation of a valid username and password with which the user can authenticate to the web application.
- Creating/editing a patient record – this is the activity of entering patient information to be recorded in the database and used later. For this operation to be possible, the user must have authenticated themselves to be able to select the correct database in which to make the record. The result of the execution of this process is a patient record in the database.
- Importing a patient file – this is the activity of entering information about multiple patients by specifying a data file in an appropriate format. The information is recorded in the database and subsequently used. For this operation to be possible, the user must have authenticated themselves to be able to select the correct database in which to make the necessary records. The result of the execution of this process is one or more patient records in the database.
- Creating a work schedule – this is the activity of choosing which hours on a weekly basis to be used in the generation of an annual schedule for preventive examinations. The choice of hours is carried out through a web form and after confirmation it is recorded in the database. For this operation to be possible, the user must have authenticated themselves to be able to select the correct database in which to make the necessary records. The result of running this process is a work schedule record in the database.

- Generating an annual schedule for preventive examinations – this is the process of automatically generating an annual schedule for preventive examinations. For this operation to be possible, 1) the user must have authenticated himself to be able to select the correct database in which to make the necessary records; 2) there must be a work schedule record based on which to select examination times; 3) there must be at least one patient record to be allocated to the schedule. The result of running this process is the creation of a new collection of time records for examination in the database.
- Implementation of an annual schedule for preventive examinations – this is the process of marking examinations as successful or postponing them to another hour. For either of the two operations to be performed, there must be an annual schedule for preventive examinations. The result of this process is an update of a patient record in the database, and if an examination is postponed, a new examination record is created.

3.3. Concept for working with the information system for generating a schedule for preventive examinations

The general concept for the implementation of an information system for generating schedules for preventive examinations is shown in Fig. 1.

When working with this application, the following basic steps should be followed:

Step 1. Registration. To be able to use the web application, registration is required. It can be done from the home page, which provides a choice - logging into the system or creating a user account. The data to be filled in is a unique username and password, which must be entered twice. Upon successful registration, the system returns to the home page, from where you can continue logging in using the newly created username and password.

Step 2. Logging into the system. Logging in to the system is carried out from the home page by entering a username and password. This step is important because, in addition to authenticating the user, a session is also carried out, through which the own database is accessed.

Step 3. Create a work schedule. This step is mandatory when a new user logs in for the first time and can be skipped in subsequent use. The work schedule is created by the user on a weekly basis, and after confirmation it is automatically transferred to all other weeks until the end of the calendar year. In the work schedule, it is mandatory to determine the days and hours for preventive examinations, and there is a possibility to fill in the GP's appointment time, but this additional information is just for convenience and is not used in any other function of the system.

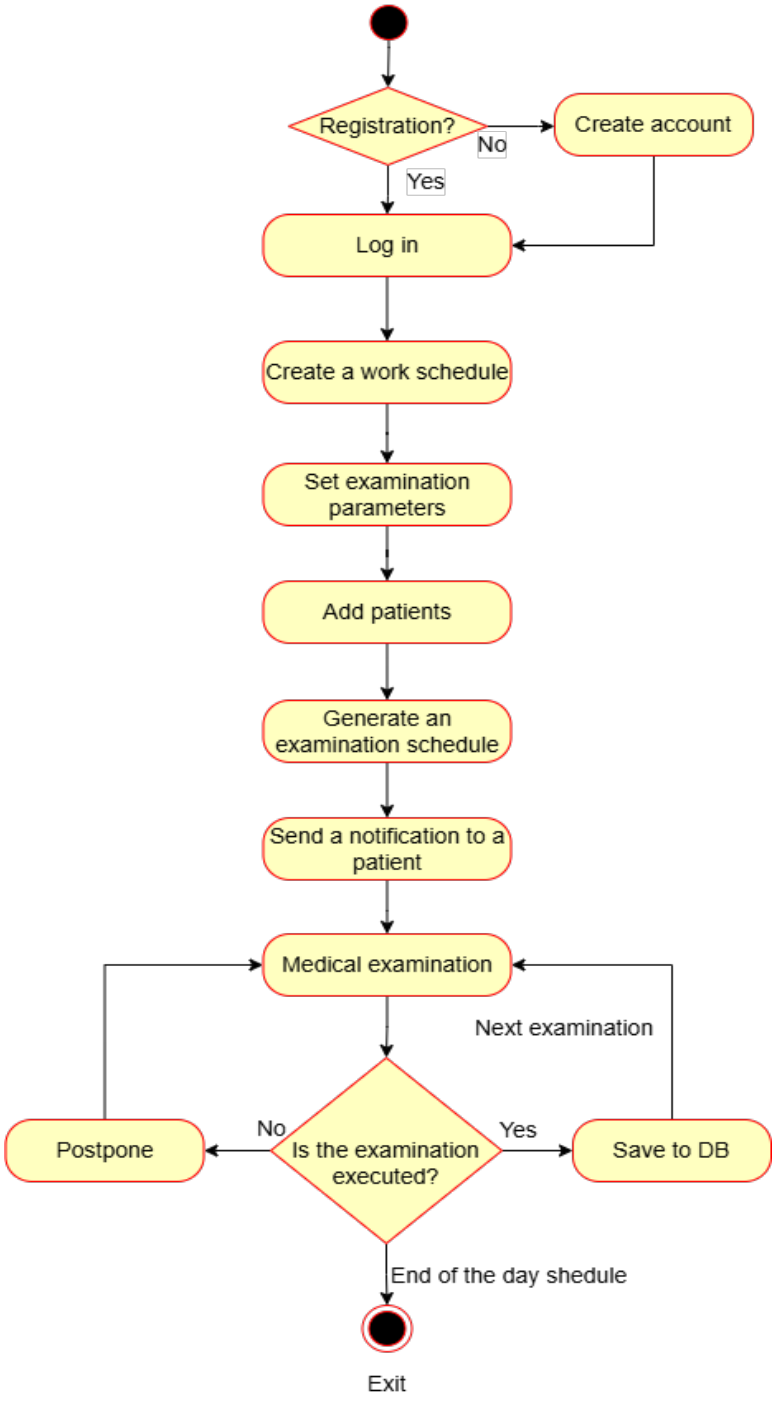

Fig. 1. General concept for working with the information system for generating a schedule for preventive examinations.

Step 4. Set examination parameters. This step must be completed before generating a schedule for a preventive examination. Here the different types of examinations are defined, as well as the criteria according to which they are set for patients. Other parameters that are set are the duration of the types of examinations and their periodicity.

Step 5. Add/edit patients. This step is mandatory only when a new user is logged in for the first time and can be skipped if there is at least one patient record in the database. Adding patients can be done in two ways:

- Adding patients from .xlsx file;
- Adding a patient by filling out a web form.

If you choose to import from a Microsoft Excel spreadsheet, it is possible to create multiple patient records automatically, but the table must be in a predefined format in the system.

Step 6. Generate a schedule for examinations. The generation of a schedule for preventive examinations can be performed when there are patients in the database and there is a work schedule. The different groups of examinations and their duration must also be set. This step can be used in two ways:

- Generation of an annual schedule – should be implemented only once within the calendar year. This action allocates an hour for examination to all patients in the doctor's work schedule.
- Generating a schedule for new patients – This action checks the patient's status and if there is still no scheduled appointment, the first possible and eligible slot within the calendar year is set.

It is possible a situation when a schedule cannot be created according to the set parameters, then there is an error message with suggestions for changing the values of key parameters, such as the duration of the types of examinations or the extension of the work schedule.

Step 7. Sending a notification to a patient. When the schedule for preventive examinations is generated, patients must be notified of their scheduled date and time. The notification can be e-mail, SMS, or both, depending on what method of communication the patient prefers. A notification can be sent individually to a single patient or to a group of patients, and the groups are determined by the time remaining until the examination – for example, all patients who have a scheduled check-up for the next month.

Step 8. Checkup (medical examination). What needs to be performed in the system during an examination is to note whether the patient has passed the examination or not.

Step 9. Record in the database. When an examination is performed, the database for the respective patient shall indicate that he/she has undergone a preventive examination for the current calendar year and the date and time of the examination shall be recorded.

Step 10. Postpone. If for some reason the examination has not taken place, it should be postponed, and the patient should be notified of the new date and time. The postponement is automatic, choosing the next possible one in the schedule for a new appointment and sending a notification to the patient.

4. Algorithm for the implementation of an information system for generating a schedule for preventive examinations

The generation of a schedule for preventive examinations is reduced to the task of distributing a certain number of patients, which is always known in advance, divided into different groups, among the vacant positions/hours of the schedule. The total number of examination hours will always be greater than the number of examinations to be distributed. It is necessary to leave spare hours to be free after the distribution of all patients, because every month changes are made to the list of people enrolled with the GP. Usually there is a balance in the number of write-offs and newly enrolled, but there is a requirement to leave a buffer of 5% of their total number of free hours for the year.

Compliance with these conditions must be ensured by checking the number of available positions before each start of the allocation algorithm, and if they are less than necessary, the algorithm does not start working until a new work schedule with enough space is created.

From a medical point of view, patients are divided into groups according to their age and the diseases they have, and different groups of patients undergo a preventive examination with different tests and manipulations. It is assumed that regardless of the differences that all types of preventive examinations have, the same duration of at least 20 minutes. For the purpose of generating a schedule, patients are divided into three groups according to their age, which differ only in what the frequency of examinations should be:

- First group of babies up to one year old – undergo a preventive examination every month until the age of one calendar year.
- Second group of children up to seven years old – undergo a preventive examination twice a year until they are 7 years old.
- Third group of adults and children over seven years old – undergo a preventive examination once a year.

Conditions for the distance in time between two consecutive examinations shall also be imposed:

- First group – a minimum of 20 days and a maximum of 40 days must have passed between two consecutive examinations.
- Second group – a minimum of 5 months must have passed and a maximum of 7 between two consecutive examinations.

- Third group – a minimum of 9 and a maximum of 15 months must have passed between two consecutive examinations.

It is important to note that a patient can move from one group to another within a single execution of the algorithm, and this action is performed under predetermined conditions. For a more visual presentation of the operation of the information system, a generalized algorithm for generating a schedule for preventive examinations is proposed, which is shown in Fig. 2.

For the description of the algorithm for generating a schedule for preventive examinations, the following notation is introduced. Each patient is presented as a composite data structure, which necessarily includes the patient's date of birth, as well as his unique identification.

A – the set of all patients less than 1 year of age;

B – the set of all patients between the ages of 1 and 7 years;

C – the set of all patients over 7 years of age;

P – the total number of patients enrolled with the GP. It is calculated as the sum of the powers of sets **A**, **B** and **C**;

PA – number of patients from group **A**. It is calculated as the power of the set **A**;

PB – number of patients from group **B**. It is calculated as the power of the set **B**;

PC – number of patients from group **C**. It is calculated as the power of the set **C**;

E – the total number of examinations that the GP must do;

W – the weekly work schedule. It must be created before the algorithm for generating a schedule for preventive examinations is launched;

date – start date for generating the schedule;

$\mathbf{t_{min}}$ – the minimum duration of one examination in minutes. The value must be set before the algorithm starts;

$\mathbf{t_{max}}$ – the maximum duration of one examination in minutes. The value must be set before the algorithm starts;

H – the total number of hours provided for preventive examinations for the year;

T – duration of one examination in minutes;

Fa() – a function that sets the hours for examination of one patient from group **A**, taking into account the possible transition to group **B** during the period for which it is generated;

Fb() – a function that sets the hours for examination of one patient from group **B**, taking into account the possible transition to group **C** during the period for which it is generated;

Fc() – a function that sets examination times for one patient from group **C**;

Fe() – a function that calculates the total number of examinations;

S – a data structure that preserves the result of the graph generation.

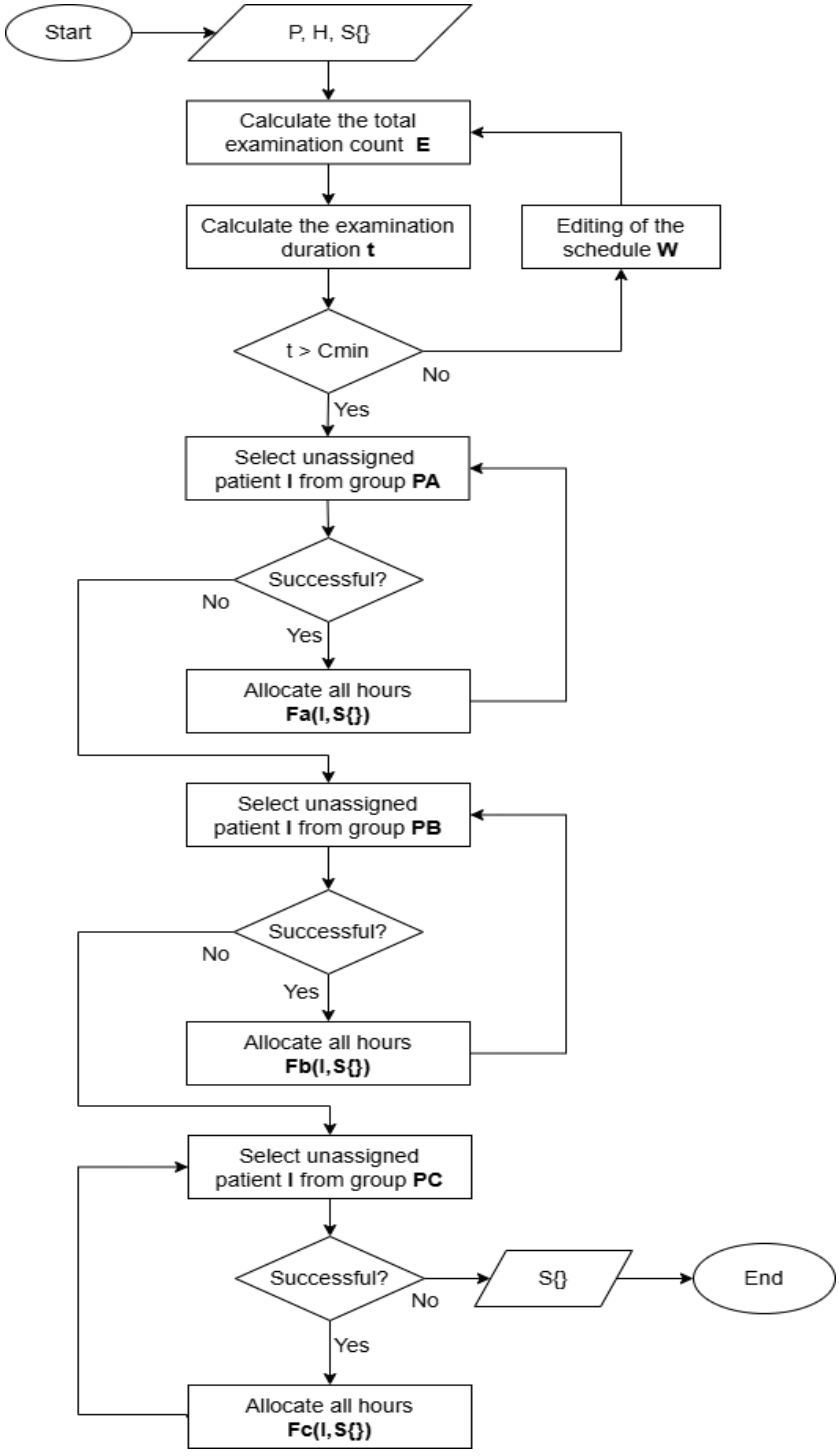

Fig. 2. Summary algorithm for generating a schedule for preventive examinations.

4.1. Function for calculating the total number of examinations - F

The total number of examinations that the GP has to perform within one calendar year is calculated by the **Fe()** function, based on the following parameters: 1) the number of patients in the three groups, 2) the date of initial generation of the schedule and 3) the date of birth of each patient, which is stored in the patient records. The listed parameters are fed as input to the function through the data structure used to describe each patient record.

The input data:

- multiple **A {1, 2, …, N}** for group **A** patients;
- multiple **B {1, 2, …, M}** for group **B** patients;
- multiple **C {1, 2, …, Q}** for group **C** patients;
- **date** - start date for generating the schedule.

Output data:

- The number of examinations in one calendar year **E**.

The pseudocode describing the algorithm of the function that calculates the total number of examinations **Fe(A, B, C, date)** is as follows:

```
A[],B[],C[];  //array of objects of patient types
date; //initial date the schedule to be generated
Fa(A, B, C, date) {
       var E = 0;
       while(a = next element from A) {
              var year = (date.year - a.year);
              var months = (date.month - a.month);
              var passedMonths = years * 12 + months;
              var remainingMonths = 12 - passedMonths;
              var EndOfYearMonths = 12 - date.months;
              if (remainingMonths < EnfOfYearMonths) {
                     if((date.months + remainingMonths) =< 6) {
                            E = E + remainingMonths +1;}
                     else {E = E + remainingMonths;}
              }
              else {E =E + EndOfYearMonths;}
       }
       while(b = next element from B) {
              if(date.month < 7 ) {E = E + 2;}
              else {E = E + 1;}
       }
       while(c = next element from C) {E = E + 1;}
       return E;
}
```

4.2. Function for allocating hours to one patient in group A

The distribution of examination hours per patient from group **A** in the schedule is carried out by the **Fa()** function. The rules by which the allocation is fulfilled depends on whether the patient has undergone examinations in the previous

schedule and, if so, on what date his last examination was. Then the examinations that are to be carried out are distributed every month until the end of the year or until all examinations are passed in the first free hour of the day, which is closest to the date of the last examination plus one month. If there is no data on the last preventive examination from the previously generated schedule, the day of birth is chosen as the day of the examination.

Input data:

- **a** data structure for a patient in group **A**;
- **date** date of schedule generation;
- **S** a data structure that stores the current schedule.

Output data:

- **S** data structure that stores the current schedule, updated with the new hours.

The pseudocode describing the algorithm of the function, which sets the hours for examination of one patient from group **A**, taking into account the possible transition to group **B** during the period for which **Fa(a, date, S)** is generated, is as follows:

```
a{};  //object of type patient record
date; initial date the schedule to be generated
S; //schedule data structure
Fa(a, date, S) {
       var flag = false;
       var E = 0;
       var years = (date.year - a.birthDate.year);
       var months = (date.month - a.month);
       var passedMonths = years * 12 + months;
       var remainingMonths = 12 - passedMonths;
       var EndOfYearMonths = 12 - date.months;
       var day = 1;
       if(a.lastExamDate == true) {day = a.lastExamDate.day;
       //last examination date exists}
       else {day = a.BirthDate.day;
              // select the day of the birth date}
       if (remainingMonths < EnfOfYearMonths) {
              if((date.months + remainingMonths) =< 6) {
                     E = remainingMonths;
                     flag = true;}
              else {E = remainingMonths;}
       }
       else {E = EndOfYearMonths;}
       while ( i < E) {
              SetExam((date.month + i), day,a,S);
       }
       if(flag = true) {
              SetExam((date.month + i + 6), day,a,S);
       }

}
```

4.3. Function for allocating the hours of one patient in group B

The distribution of the examination hours of group **B** patients in the schedule is carried out by the function **Fb()**. This feature should distribute one or two examinations per year as evenly as possible. If the start date for a schedule generated is in the first half of the year, then there should be two examinations, unless in the corresponding period between the two examinations the patient moves to group **C**. If the date for generation is in the second half of the year, the examination must be one.

Input data:

- **b** data structure for group **B** patient;
- **date** - date of schedule generation;
- **S** a data structure that stores the current schedule;
- **S** data structure that stores the current schedule, updated with the new hours.

The pseudocode describing the algorithm of the function, which sets the **hours** for examination of a patient from group **B**, taking into account the possible transition to group **C** during the period for which **Fb(b, date, S)** is generated, is as follows:

```
Fb(b, date, S) {
       var E = 0;
       var day = 1;
       var month = 1;
       if(date.month > 6 ) {
              if (b.lastExamDate == true) {
                     day = a.lastExamDate.day;
                     month = a.lastExamDate.month + 6;}
              else {day = 15;
                     month = (12 - date.month) % 2;}
              if (SetExam(month, day, b, S) {return OK}
              else if (SetExam(month +1, day, b, S) {return OK}
              else if (SetExam(month -1, day, b, S) {return OK}
              else {return Err}
       }
       else {
              if((date.year - b.year)*12 + (date.month - b.month)
> (7*12+6)) {var iterations = 1;}
              else {var iterations = 2;}
              var i = 0;
              if (b.lastExamDate == true) {
                     day = a.lastExamDate.day;
                     month = 6 - (12 - a.lastExamDate.month);}
              else {day = 15;
                     month = (6 - date.month) % 2;}
       while (i < iterations) {
              month = month +(6* i);
              if (SetExam(month, day, b, S) {return OK}
              else if (SetExam(month +1, day, b, S) {return OK}
```

```
                else if (SetExam(month -1, day, b, S) {return OK}
                else {return Err}
        }
}
```

4.4. Function for allocating hours to one patient in group C

The distribution of the examination hours of group **C** patients in the schedule is carried out by the **Fc()** function, which sets the hours for examination of one patient from group **C**. This function must allocate one examination for the whole year. If a record exists for the last examination, the feature should schedule the new one for the closest date to twelve months after the last examination.

Input data:

- **c** data structure for a patient in group **C**;
- **date** - date of schedule generation;
- **S** a data structure that stores the current schedule.

Output data:

- **S** data structure that stores the current schedule, updated with the new hours.

The pseudocode describing the algorithm of the function **Fc(c, date, S)** is as follows:

```
Fc(c, date, S) {
        var day = 1;
        var month = 1;
        if (c.lastExamDate == true) {
                day = c.lastExamDate.day;
                month = c.lastExamDate.month;
                j = 0;
                k = 0;
                while (true) {
                        k = k +1;
                        if (k % 2 == 1) {j = j +1;}
                        if((month > 0) && (month =< 12)) {
                                if (SetExam(i, day, c, S) {
                                        return OK;}
                        }
                        else {
                                month = month +((-1)^j)*j;
                                if ((abs(j) > 3) {
                                        return Err;}}
                }
        }
        else {
                var i = 1;
                while (i < 12) {
                        if (SetExam(i, day, c, S) {return OK}
                }
        }
}
```

4.5. Patient Appointment Function

The function of setting an appointment for a patient examination is auxiliary and is used by the functions **Fa()**, **Fb()** or **Fc()**. It records a time in a certain month and day, and if there are no vacancies on the desired day, then it finds the nearest free time on another day. The feature searches for another day only within the month for which it was launched. The way it works is by trying to write an hour in the structure of the schedule for a selected month and day, and if this operation is unsuccessful, it tries for the next day, if it fails again - for the previous one, then for two days ahead and two days back, etc. The operation that performs this action depends on the selected programming language and the data structure for storing the schedule.

Input data:

- **month;**
- **day;**
- an element of the set of all patients **Patient**;
- a data structure that stores the current schedule **S.**

Output data:

- The result of the success/error performance.

The pseudocode describing the algorithm of the **SetExam function (mon,day,Patient,S)** is as follows:

```
SetExam(month,day,Patient,S) {
       daysInMonth = the number of days in the month (month);
       var j = 0;
       var k = 0;
       while (true) {
              k = k +1;
              if (k % 2 == 1) {
                     j = j +1;
              }
              if((day > 0) && (day =< daysInMonth)) {
                     if (S.SelectSlot(a, day, month) == true) {
                            return Successfully selected slot;
                     }
              }
              else {
                     day = day +((-1)^k)*j;
                     if ((abs(j) > daysInMonth) {
                            return The function failed wit error
                                   No free slots;
                     }
              }
       }
}
```

5. Conclusion

The article presents a structured elicitation of requirements that address the required functionality and the diverse aspects of the environment where the information system for the generation schedule for preventive examinations will operate. The proposed algorithmic realization of the information system represents a significant step toward a more proactive healthcare environment, where patients and healthcare providers can work in tandem to ensure timely, preventive care. For this goal, an algorithm for schedule generation is proposed considering the requirements and constraints. The proposed pseudo-codes realized different functionalities enable flexible technology choices, accommodating future scalability and functionality. The developed demo version of the system validated the business logic, with numerical testing confirming the feasibility of a fully functional solution.

The flexibility of the system's design allows for adaptations to meet the evolving needs of healthcare practices, ultimately helping to streamline workflows, improve patient outcomes, and enhance overall efficiency in healthcare service delivery.

Implementing the proposed information system for schedule generating and organizing mandatory preventive examinations contributes to timely reminders for visits to their GP.

Potential future developments include enabling patient accounts for online schedule access; expanding functionalities to create a complete work schedule and manage appointment slots; and evolving the system into a comprehensive patient management platform.

References

1. Jeong, H.S., Chung, H.: Bridging smart technologies and healthy cities: A scoping review using WHO's 6P framework, Sustainable Cities and Society, 116, 105888, 2024, https://doi.org/10.1016/j.scs.2024.105888.
2. Borissova, D., Dimitrova, Z., Garvanova, M., Garvanov, I., Cvetkova, P., Dimitrov, V., Pandulis, A.: Two-stage decision-making approach to survey the excessive usage of smart technologies. Problems of Engineering Cybernetics and Robotics, vol. 73, pp. 3–16 (2020), https://doi.org/10.7546/PECR.73.20.01.
3. Borissova, D., Keremedchieva, N.: Decision support approach in evaluating the parameters of books for digital manufacturing. In: Rocha, Á., Ferrás, C., Méndez Porras, A., Jimenez Delgado, E. (eds) Information Technology and Systems. ICITS 2022. Lecture Notes in Networks and Systems, vol. 414, pp. 165–174 (2022), https://doi.org/10.1007/978-3-030-96293-7_16.
4. Borissova, D., I. Mustakerov. An integrated framework of designing a decision support system for engineering predictive maintenance. Int. Journal Information Technologies & Knowledge, vol. 6(4), pp. 366–376 (2012).

5. Borra, P.: An Overview of Cloud Computing and Leading Cloud Service Providers (May 29, 2024). International Journal of Computer Engineering and Technology, vol. 15(3), pp. 122–133 (2024), https://doi.org/10.17605/OSF.IO/5HQ4M.
6. Tang, Z., Tang, Z., Liu, Y., Tang, Z., Liao, Y.: Smart healthcare systems: A new IoT-Fog based disease diagnosis framework for smart healthcare projects, Ain Shams Engineering Journal, vol. 15(10), 102941 (2024), https://doi.org/10.1016/j.asej.2024.102941.
7. Borissova, D., Keremedchieva, N., Keremedchiev, D.: Business intelligence approach to support decision making in publishing sector, In: 43rd International Convention on Information, Communication and Electronic Technology, Opatija, Croatia, 2020, pp. 1268–1273, https://doi.org/10.23919/MIPRO48935.2020.9245424.
8. Borissova, D., Mustakerov, I.: A concept of intelligent e-maintenance decision making system, In: IEEE INISTA, Albena, Bulgaria, 2013, pp. 1–6, https://doi.org/10.1109/INISTA.2013.6577668.
9. Garvanova, M., Garvanov, I., Borissova, D.: Computer model for assessment and visualization of specific absorption rate of electromagnetic field, generated by smartphone. In: Shishkov, B. (eds) Business Modeling and Software Design. Lecture Notes in Business Information Processing, vol. 453, pp. 299–307, 2022, https://doi.org/10.1007/978-3-031-11510-3_21.
10. Taylor, R.N.: Software Architecture and Design. In: Cha, S., Taylor, R., Kang, K. (eds) Handbook of Software Engineering, Springer, Cham (2019), https://doi.org/10.1007/978-3-030-00262-6_3.
11. Zhukovska, A., Brechko, O., Zheliuk, T., Olga, C., Shushpanov, D., Olena, N.: Information system and technologies in the health care management. In: 12th Int. Conference on Advanced Computer Information Technologies, Ruzomberok, Slovakia, 2022, pp. 249–254, https://doi.org/10.1109/ACIT54803.2022.9913132.
12. Koleva, I., Yoshinov, B., Yoshinov, R. Digitalization of educational materials and establishment of electronic archive in the rehabilitation field. Digital Presentation and Preservation of Cultural and Scientific Heritage, vol. 12, 2022, pp. 277–284, https://doi.org/10.55630/dipp.2022.12.26.
13. Lv, Zh., Qiao, L.: Analysis of healthcare big data. Future Generation Computer Systems, vol. 109, pp. 103–110 (2020), https://doi.org/10.1016/j.future.2020.03.039.
14. Borges do Nascimento, I.J., Abdulazeem, H., Vasanthan, L.T. et al. Barriers and facilitators to utilizing digital health technologies by healthcare professionals. npj Digit. Med. 6, 161 (2023), https://doi.org/10.1038/s41746-023-00899-4.
15. Boneva, A., Boneva, Y.: An approach for encrypted exchange of information in corporate networks based on Tcl/Tk. Problems of Engineering Cybernetics and Robotics, vol. 78, pp. 5–22 (2022), https://doi.org/10.7546/PECR.78.22.02.
16. Gaidarski, I.: Model driven development of information security system. Problems of Engineering Cybernetics and Robotics, vol. 76, pp. 47–62 (2021), https://doi.org/10.7546/PECR.76.21.04.
17. Mustakerov, I., Borissova, D.: Data structures and algorithms of intelligent web-based system for modular design. (2013), https://doi.org/10.5281/ZENODO.1086885.