Open Access

Open Journal of Information Systems (OJIS)
Volume 3, Issue 1, 2016

www.ronpub.com/ojis
ISSN 2198-9281

A NoSQL-Based Framework for Managing Home Services

Marinette Bouet, Michel Schneider

LIMOS, Clermont-Ferrand University, Complexe des Cézeaux, 63178 Aubiere, Cedex, France,
marinette.bouet@univ-bpclermont.fr, michel.schneider@isima.fr

ABSTRACT

Individuals and companies have an increasing need for services by specialized suppliers in their homes or premises. These services can be quite different and can require different amounts of resources. Service suppliers have to specify the activities to be performed, plan those activities, allocate resources, follow up after their completion and must be able to react to any unexpected situation. Various proposals were formulated to model and implement these functions; however, there is no unified approach that can improve the efficiency of software solutions to enable economy of scale. In this paper, we propose a framework that a service supplier can use to manage geo-localized activities. The proposed framework is based on a NoSQL data model and implemented using the MongoDB system. We also discuss the advantages and drawbacks of a NoSQL approach.

TYPE OF PAPER AND KEYWORDS

Regular research paper: *Home services, DSL, framework, NoSQL, MongoDB, healthcare, cleaning services*

1 INTRODUCTION

In this paper we are interested in activities performed at private homes as well as activities performed at commercial locations by a service provider. The approaches for specifying, planning and achieving such activities are similar regardless of the nature of these activities. Consequently, our objective here is to propose a common framework to support their management.

In-home services for private individuals represent a business sector that has become increasingly important in all the western countries. The types of activity vary widely and include housework, childcare, elder care, short- and long-term care for handicapped and disabled people, landscaping services and home medical care. However, it is the home medical care sector that is the most organized and requires the most attention. The term “home care” as used here includes care for sick or dependent people, disabled adults and the chronically ill. In France, home medical care services are supplied by licensed organizations such as SSIAD (home care services), HAD (hospital care at home) and SAMSAH (medical and social services for handicapped adults). Healthcare is regulated and must adhere to strict legal and medical constraints. Moreover, persons involved in the at-home care (usually professionals, but may also be parents or close relatives) must comply with statutory conditions and must have the requisite skills.

For these home-care interventions, the main problem is optimal organization based on the care required and the available resources [15]. Many previous studies have investigated software tools to help manage home care activities [10]. On one hand, some of these studies propose adapting ERP software, primarily to meet classic management needs such as

patient management, admissions and discharges, keeping patient records and managing billing and invoicing tasks. On the other hand, other studies have designed and implemented diverse algorithms to help in planning interventions and allocating human resources. The difficulty at this level is to integrate the various constraints. These two types of works are far too often treated separately.

Services offered at company sites have represented an expanding sector for quite some time. Even in this field, the activities required are varied: arranging and maintaining the premises, cleaning, equipment installation and maintenance, repairs, training, workplace assistance, etc. To explore this topic, we will focus primarily on cleaning services, the management of which is very close to management of home care and involves similar steps: determining services, planning interventions, allocating human resources, monitoring service delivery, and so on. Commercial ERP software tools have been specifically designed for this sector to help cleaning agencies to organize their activities.

To our knowledge, no unified approach exists to handle similar situations for different industry sectors. This paper's main objective is to propose a generic framework that allows service providers to manage geo-localized activities regardless of the nature of their services. Another objective is to design this framework so that it can be both easily extended and easily personalized. This framework is suitable for any type of activity that repeats fairly regularly and whose duration ranges from a few minutes to a few hours per day. Our proposal handles data using a NoSQL system [12] and initiates processing via web services. A NoSQL approach presents several advantages over other technologies: it is easier to structure the data to specific needs; the design is closer to the implementation and development is simple; and it is easier to evolve data structures and modify processing. Nonetheless, this approach suffers from some drawbacks, including difficulty expressing integrity constraints, installation difficulty and lack of transactional control. We shall show how to minimize the consequences of these difficulties.

In addition to this introduction, the paper includes 6 other parts. Part 2 describes the current state of the art. Part 3 presents the general characteristics of the proposed framework and its architecture. Part 4 details the conceptual model for managing activities. Part 5 describes the logical model based on a NoSQL representation of documents. Part 6 describes an implementation using MongoDB. Finally, Part 7 contains our conclusion and future perspectives. Appendix A describes the syntax for our temporal language.

2 RELATED WORK

For a long time, the home care and cleaning services sectors occupied the attention of professionals, who have marketed numerous software tools for managing those activities. For example, in an Anglo-American context, the authors of [10] list 10 tools recommended for managing medical activities, and the authors of [9] list 40 tools for managing activities in the cleaning sector. The core functionality is rather similar on both sides and contains a variety of features such as billing and invoicing, client management, mobile access, GPS integration, scheduling and routing, human resource management, quality improvement, etc. Approached from a French context, similar tools exist, in particular AtHome [1] and MedLink [22] for home care management. All these tools are expensive - often well out of the purchasing range of a small company. To our knowledge, no comparative detailed studies exist to support an informed choice among such tools, and it is quite difficult to obtain information on the logical architecture or the technologies these tools use.

Researchers were especially interested in the sector of home care. We can distinguish two main thrusts of the research: modelling activities and planning/routing services.

There are several main works that are concerned with modelling activities. In [17], a multi-agent system is proposed to enhance monitoring, surveillance and educational services for chronic disease management. In [21], a framework is proposed for understanding home care operations and their management. The main processes are identified, and an IDEF0 description is suggested. In [18], the authors state that workflows provide an adequate means of supporting coordination and monitoring of care processes. Because care processes depend directly on patient profiles, an ontology-driven approach is proposed to help with the construction of personalized careflows.

In [6], the problem of improving the cooperation between actors through a cognitive approach is addressed; it specifies the requirements for cooperation and proposes a framework and a prototype. In [23], the concepts of BCP (Business Continuity Plan) and MDE (Model Driven Engineering) are applied in the context of e-Health systems. The Plas'O'Soins project [3], to which we contributed, is centred on the coordination and follow-up of home care activities. A prototype was developed whose architecture was based on three main modules: a specification module for care description, a planning and routing module and an intervention recording module. An experiment was performed in conjunction with three home care institutions.

The planning of home care and the optimal determination of tours has been the object of many

works, so bibliography is plentiful on the subject. We shall focus on some important points. First, the proposals differ by the constraints that were taken into account. These constraints are multiple and concern the caregivers (availability, jobs and qualifications, types of contracts, preferences of patients, tour localization, etc.), the patients (required qualifications, schedule preferences, unavailability, caregiver preferences, etc.), the care institution (respect for caregivers' work schedules, caregivers' preferences for well-balanced tours and caregivers' preferences for full employment, etc.), and the medical practices (specifically, dependences between activities). A description of some of these constraints appears in [5].

The objective of these studies—which function to optimize—is not always the same and includes minimizing the operating costs (by considering staff costs) [19], minimizing the number of trips, minimizing the drive time for all tours and minimizing total tour duration. This type of problem is characterized by the abbreviation MTSPTW (Multiple Travelling Salesman Problem Time Windows). In some cases, the problem can be formulated in the form of a linear program, and some studies proposed exact approaches. However, considering its NP complexity, researchers have also studied how to resolve the problem using heuristic approaches. Table 1 lists an overview of some of these approaches.

These various research works are essentially theoretical in nature. Most were validated using only theoretical cases or synthetic data. Nevertheless, considering the complexity and the variability of the problems, an experiment in a real-world context represents an undeniable gain. The experiment carried out in the Plas'O'Soins project [3] was instructive and incited the researchers to concentrate on specific issues. Specifying activities and planning surfaced are the most important points.

Table 1: Approaches for planning and routing

Exact approaches	
Overview	Bektas [4]
Extended Multiple Traveling Salesman Problem	Kergosian et al. [15]
Multiple Treatments and Time Windows	Torres Ramos et al. [25]
Heuristic approaches	
Tabu search	Gourc et al. [11] Jemai et al. [13]
Genetic algorithms	Kiraly et al. [16]
Ant colony optimization	Liu et al. [20]
Rules engine	Weppenaar et al. [26]

3 THE POSITIONING OF THE FRAMEWORK

In this section we list the principles on which our proposal is based and detail the architecture and the functions of our framework.

3.1 Characteristics and objectives

We position our proposals in a unified framework that fulfils the following characteristics.

C1: The framework is calibrated to serve the needs of a service provider which collects resources to perform the services.

C2: The services are the object of a contract or a preliminary agreement.

C3: A service is exactly geo-localized, either to a home or to customer premises. Thus, it is necessary to physically move resources around to perform the services.

C5: A service is composed of activities. An activity respects a precise timing defined by temporal intervals (scheduling start times as soon as possible, scheduling end times as late as possible) with more or less regular repetitions.

C6: The activities are predefined and listed in a catalogue. For each activity, this catalogue defines the types of staff able to perform it, the standard duration of the activity and the price unit.

C7: A staff member may be permanent or temporary (professional person in the case of a home care institution, temporary person in the case of a cleaning agency).

C8: The activities require equipment provided by the institution or the agency and consumables that must be supplied on demand. Equipment must be preinstalled and regularly maintained.

C9: An activity requires, at minimum, one staff member with the necessary skills; if several staff members are required, at least one must possess the necessary skills and be appointed as the main staff member.

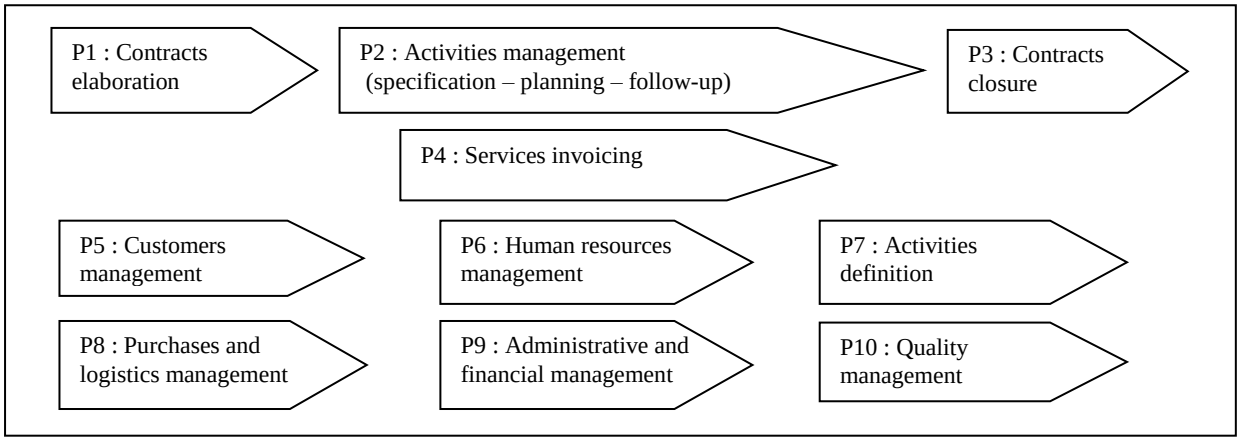

Figure 1: Main processes of the proposed framework

The various objectives that this framework must guarantee are listed below.

O1: Specify the interventions to be made at each customer location according to previously agreed upon commitments or contracts.

O2: Organize the planning of the interventions by selecting the duration and the necessary resources for each activity, using these values to subsequently plan tours for permanent staff members.

O3: Support service follow-up and logging of every malfunction or problem to help with finding solutions.

O4: Handle service invoicing, purchasing, delivery of consumables, equipment purchases and/or rentals installed for customers, vehicle purchase and/or rental, salaries for permanent staff, remuneration for temporary staff, etc.

O5: Establish balance sheets for activities to allow evolution of the practices to improve the quality of the services.

O6: Continually assess activities and allow for modifications concerning the evolution of the service providing institution or agency such as defining new services, prospecting for new customers, recruiting new staff members, etc.

3.2 Organization of the Processes

The main processes that fulfil objectives O1 to O6 are represented in Figure 1. The heart of the framework is constituted by processes P1, P2, and P3, which allow for elaborating service contracts, managing the activities corresponding to these services, and contract termination, respectively.

In home care, P1 corresponds to all the preliminary operations required to decide whether to accept a new patient and determine that patient's care protocols. P3 supports all the procedures required to end a home care contract. Similarly, for a cleaning agency, P1 corresponds to acquisition of a new customer and defining services associated with that customer's contract. P5 deals with managing all the customer information required for the activities. In home care, that information consists of a patient's physical characteristics and medical records. For a cleaning agency, it consists of information about the state and arrangement of the customer's premises. In both cases, storing the geographical address of the home or the premises is required for precise estimation of travel time and for calculating the tours. P7 is the process that defines the various activities and the skills of the staff members required to perform them according to the contract. In the rest of this paper, we shall concentrate on the process P2, which coordinates the main operations.

4 CONCEPTUAL MODELLING OF ACTIVITIES MANAGEMENT PROCESS

This section is both a generalization and an extension of the work which we previously led on the Plas'O'Soins project [8].

4.1 Organization and Operation

Execution of the activities for the various customers is supervised through the process P2, which deals with activities management. This process consists of three sub-processes: activity specification, activity planning and execution and follow-up (Figure 2).

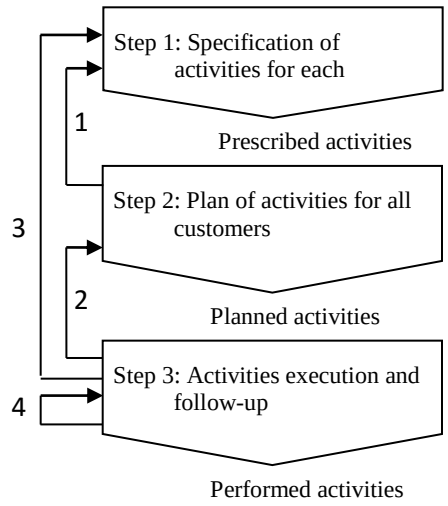

Figure 2: The three sub-processes of the activities management process

Step 1: Specifying the activities associated with each customer is the initial sub-process, and this sets conditions for the other two. This process begins as soon as the contract process with a customer concludes.

Step 2: Activity planning can be a continual process depending on the business needs. It consists of identifying the beginning and end of each intervention and assigning one or more staff members to the activity, while also taking into account travel times and the duration of the interventions. This process determines the daily tours for every permanent staff member. It is easy to see how most businesses would need to run this planning process every day or every week. The activities to be completed must have been specified previously for the entire period covered by the planning process (for example, day or week). The planning must satisfy various types of constraints (cf. section 2). We can envisage strong constraints as those that must necessarily be satisfied, while weak constraints are those that should be satisfied if possible. Using an automated software tool to calculate the planning leads to some limitations because it is difficult to take all aspects of the problem into account. However, automation can supply a significant amount of help. It is imperative to design an interface that allows users to finalize the solution.

Step 3: The sub-process of execution-follow-up allows every staff member to refer back to information relevant to each of the interventions made during a tour and to record information about the results.

Typically, the three sub-processes run in sequence. Feedback must analysed to solve any problems encountered during execution. We list below the most important problems (the numbers in the text refer to Figure 2).

1. If the automatic tool is unable to find a planning solution, the activities specification must be altered and rescheduled; alternatively the coordinator must complete the plan manually.

2. A problem detected at the execution level may require a modification of the current planning but still use the same activities.

3. A problem detected at the execution level may require modification of the corresponding activities. A new planning process must then be completed, either at the level of the activities concerned (a local rescheduling operation) or at the level of all the activities (a global rescheduling operation). The system can first attempt a local rescheduling; if that fails, it can launch a global rescheduling operation.

4. A problem detected at the execution level can raise warnings for the continuation of the corresponding activity; however, it is not necessary to modify it or to reschedule it.

This decomposition in three sub-processes infers three views of the activities—prescribed activities, planned activities and completed activities—all of which must be managed simultaneously. Data capture and reporting are handled through two main interfaces: the coordinator interface and the staff member interface. Activity specification and planning are handled through the coordinator interface, and execution follow-up is handled through the staff member interface.

4.2 Temporal Modelling

The activities in which we are interested here are repetitive with more or less regularity. An explicit specification of the repetition schedule is required to prepare the planning. This specification can be performed using a calendar (developed form), but then repetitions do not appear directly. Another solution is to specify schedules in the form of assertions (condensed form) where repetitions are expressed directly in a semi-natural language (e.g., daily at 8:00 except Sunday). The condensed form of specification is closer to common medical usage, is easier to communicate and allows for more sophisticated reasoning than the developed form. In this section, we propose a general model for expressing almost-regular repetitions in condensed form. Irregularities show up for at least two reasons: increased activity in certain periods and exceptions on certain days.

The model presented below is an improvement from the model [7] we suggested for the Plas'O'Soins project. We assume that an activity to be specified takes place during a given period characterized by a starting date and an ending date. If the period is not specified, then the activity begins at the date of the specification (upon validation) and continues until a new specification introduces a conflict. Within the specified period, the activity repetition is expressed in the days that compose it. Within a day, an activity can occur within one or several disjoint intervals. We call temporality the triplet (period, days, intervals) and use expressions to specify its different forms.

A formal description of these expressions is given in Appendix A. Here we provide only an intuitive presentation that conforms to a user's view. Days can be a list of weekdays, a list of dates or an expression (every day, every (2) days, odd days, even days, holidays, etc.). Intervals may contain several values, each one representing a time window: a part of the day (i.e., morning, evening, etc.), a time slot (i.e., 10 h-11 h) or a specific time (i.e., 10 h), which means that the activity should start precisely at this time. All these values can be combined as long as the corresponding time windows do not overlap. Finally, in some cases, it is sufficient to simply indicate the number of times an activity must take place within one day (without specifying time windows). We will illustrate this model using the cases listed in Table 2.

Case 1 and Case 2 introduce simple repetitions. For Case 1, the activity takes place on certain days of the week in the morning and the evening. For Case 2, the activity takes place every day in the morning. However, the repetition of an activity is not always perfectly regular. Furthermore, there may be exceptions. Our language allows users to combine several expressions to specify overlapping repetitions or exceptions. Such combinations pose no problem as long as the days associated with the expressions of a same activity are disjoint and in pairs, as in Case 3. However, a problem arises when the days associated with different expressions are not disjoint, as shown in Case 4. This case is ambiguous because it is unknown whether Sunday activities must take place in the morning and in the evening or in the morning only. To resolve this ambiguity, we introduce the notion of exceptions via the keyword "except". Thus, the above case is correctly specified, as indicated in Case 5. Case 6 introduces exceptions on Sunday (the activity takes place in the morning only) and on holidays (no activity occurs on a holiday because a holiday is not associated with a specification).

A correct specification must satisfy the following two conditions: i) the intervals associated with an expression must not overlap, and ii) the days designated by the different expressions of the same activity must not overlap.

Table 2: Good and bad specifications for a repetition

Case		Activity	Period	Days	Intervals
1	OK	Toilet	04/21/13-05/15/13	Monday, Wednesday, Friday	Morning, evening
2	OK	Toilet	04/21/13-05/15/13	Everyday	Morning
3	OK	Toilet	04/21/13-05/15/13	Monday, Wednesday, Friday	Morning, Evening
			04/21/13-05/15/13	Sunday	Morning
4	Ambiguous	Toilet	04/21/13-05/15/13	everyday	Morning, Evening
			04/21/13-05/15/13	Sunday	Morning
5	OK	Toilet	04/21/13-05/15/13	Every day except (Sunday)	Morning, Evening
			04/21/13-05/15/13	Sunday	Morning
6	OK	Toilet	04/21/13-05/15/13	Every day except (Sunday, holidays)	Morning, Evening
			04/21/13-05/15/13	Sunday	Morning

4.3 Specification of Different Granularities for an Activity

The specification of an activity must be as flexible as possible and able to adapt to different situations that may be encountered. Therefore, we suggest three levels of granularity for specifying an activity.

Fine granularity. This level corresponds to the most precise specification and should be used whenever the activity must be executed according to a specific procedure (defined in the catalogue).

```
Patient: DUPONT Gérard
Activity: Toilet in washbasin
Period: 04/21/13-05/15/13
Days: every day except holiday
Intervals: morning
Number of staff: 1
Duration: 10 minutes
Staff type: Nursing assistant
Comments: Warning, the patient has
paralysis in his right arm.
```

Listing 1: An activity with a fine granularity

Listing 1 shows a typical example in the context of home care. The operations at this level are usually well defined; the specified duration will be respected, and the performed schedule will often be close to the theoretical schedule.

Middle granularity. This granularity corresponds to the specification of a general activity. Some variability exists in the execution, which depends on the conditions encountered (state of the patient, degree of soiling of a room).

```
Patient: AUDIARD Jules
Activity: Toilet
Period: 04/21/13-05/15/13
Days: every day except holiday
Intervals: morning
Number of staff: 1
Duration: 20 minutes
Staff type: Nursing assistant
Comments: This patient has difficulty
getting up. According to the patient's
state, choose a toilet at a washbasin
or a toilet in bed.
```

Listing 2: An activity with a middle granularity

The modalities of achievement are indicated in comments. These are only recommendations because it is not possible to foresee the exact operations to make. The example in Listing 2 illustrates the middle level of granularity. In some cases, the observed duration may be significantly different from the specified value. The performed schedule will may diverge from the theoretical one.

Coarse granularity. At this level it is simply stated passages at home. Each required passage is specified by its temporalities and an indicative duration. In the medical context, regular passages are often necessary to prevent any worsening of the patient's state. We illustrate this in Listing 3. Potential treatments to be performed are listed in the Comments field.

```
Patient: DUMONT Aurélie
Activity: Nurse passage
Period: 04/21/13-05/15/13
Days: every day except holiday
Intervals: morning
Number of staff: 1
Duration: 30 minutes
Staff type: Nurse
Comments: This patient has difficulty
breathing. Control the temperature
and the blood pressure. Install if
necessary oxygen mask. In the
morning, if the patient's condition
allows it, make him run a few steps.
```

Listing 3: An activity with coarse granularity

Each passage can be personalized. For example, the recommendations for the patient Aurélie DUMONT for the morning passage could be different from those of the evening passage. These recommendations will be reported in the "comments" field. Another solution is to introduce two different passage specifications. The duration of a passage is merely indicative; the performed schedule may be very different from the theoretical one.

In general, the higher the granularity level is, the more the performed schedule differs from the theoretical one. It is important to note that the three levels of granularity can be managed by a single data structure and a single interface specification (see Sections 5 and 6).

4.4 Grouping Activities and Generating Interventions

To calculate the planning from the specifications, we must first transform temporal condensed forms (see section 3.4) in developed form. It is indeed necessary to determine the activities to be carried out for each day

of the planning horizon. This transformation can be easily performed by a simple algorithm exploiting the features of our temporal language. Moreover, before making the planning, it is advantageous to group all activities that can be performed for a given customer in the same time interval by the same type of staff member. This will minimize the number of trips. This grouping must be made for each customer and for each day. For example, for the patient AUDIARD Jules (Listing 2), suppose that a head cleaning is required every Saturday morning. On Saturday morning, this activity can be grouped with the toilet specified in Listing 1.

A grouping of activities for a given customer and a given day, must respect the following principles:

(i) The activities of a group must all be carried out by the same type of staff member.

(ii) The activities of a group must run in sequence without waiting (we can tolerate in some cases a small waiting period) in respect of their durations and their intervals.

(iii) The activities of a group must not conflict with an activity of another type of staff member.

Several types of algorithms can be devised to make the grouping. Constraints can be introduced to avoid the production of too large groups that would not be appropriate. For example, in home care, it is necessary to avoid too long presence of a staff member with a patient. It is the grouping algorithm which determines the duration and the time interval of an intervention from the duration and the time interval of each activity which composes it. Such algorithms must be regarded as a help and users must manually adjust the resulting grouping. At the end, each group represents an intervention in the patient's home or at the customer's premises. These are the interventions which constitute the entry point for the planning procedure. The grouping concept so defined may be considered at any level of granularity. Thus, we can group a specific activity with a passage if their temporal specifications match. This means that, during the passage, the staff member perform an supplementary operation in addition to those that were required for the activity.

4.5 Planning Activities

Planning has to determine the schedule of interventions according to the availability of permanent staff members. When overload conditions occur, it is appropriate to involve temporary staff. In our framework, we suggest a procedure using two main steps: determination of tours and assigning interventions to tours.

Step 1. *Determination of daily tours*: The number of tours for each planning day is determined based on the availability of permanent staff members. This determination must consider the number of working hours each staff member must achieve, the work schedules of the business, overtime work and due holidays. The output of this step is a set of possible tours in a day, each of which lists the start time, the end time and the available staff members. A staff member can complete several tours in a day (for example, one in the morning and one in the evening). This step can be partially automated.

Step 2. *Assignment of interventions to tours:* The assignment of interventions to tours involves placing each intervention on a tour with respect to the type of staff member, the expected duration of the tour and the specified time interval. The system can seek to optimize one or more criteria, for example, by maximizing the number of assigned interventions, minimizing travel time, minimizing waiting time or minimizing the total duration of tours. The constraints to be satisfied may be more or less complex (see Section 2). Some constraints are strong (mandatory to satisfy); others are weak (satisfy if possible). As stated in Section 2, many studies have focused on this issue in the context of home care. Efficient algorithms are now available and can be adapted to a given context. However, it is difficult to take all the constraints of the real problem into account. We must therefore consider this type of algorithm as an aid. A user must validate the result and adjust it manually if required.

Interventions not assigned to a tour should be performed by temporary staff. If necessary, it is possible to assign priorities to interventions so that high-priority operations are always assigned to a tour.

4.6 Realization and Follow-up of Activities

It is important to capture the results of each completed intervention, however briefly, to enable monitoring of services and their continuous improvement. The preparation of this report can be simplified by using predetermined forms. With the advent of digital technology, it can be very useful to equip each staff member with a smartphone or a tablet. It then becomes possible for each staff member to retrieve information about the planned tour before departing, to create the report in real time, to send it after each intervention. The format of the reporting form may be pre-adapted to correspond to the intervention by the planning procedure.

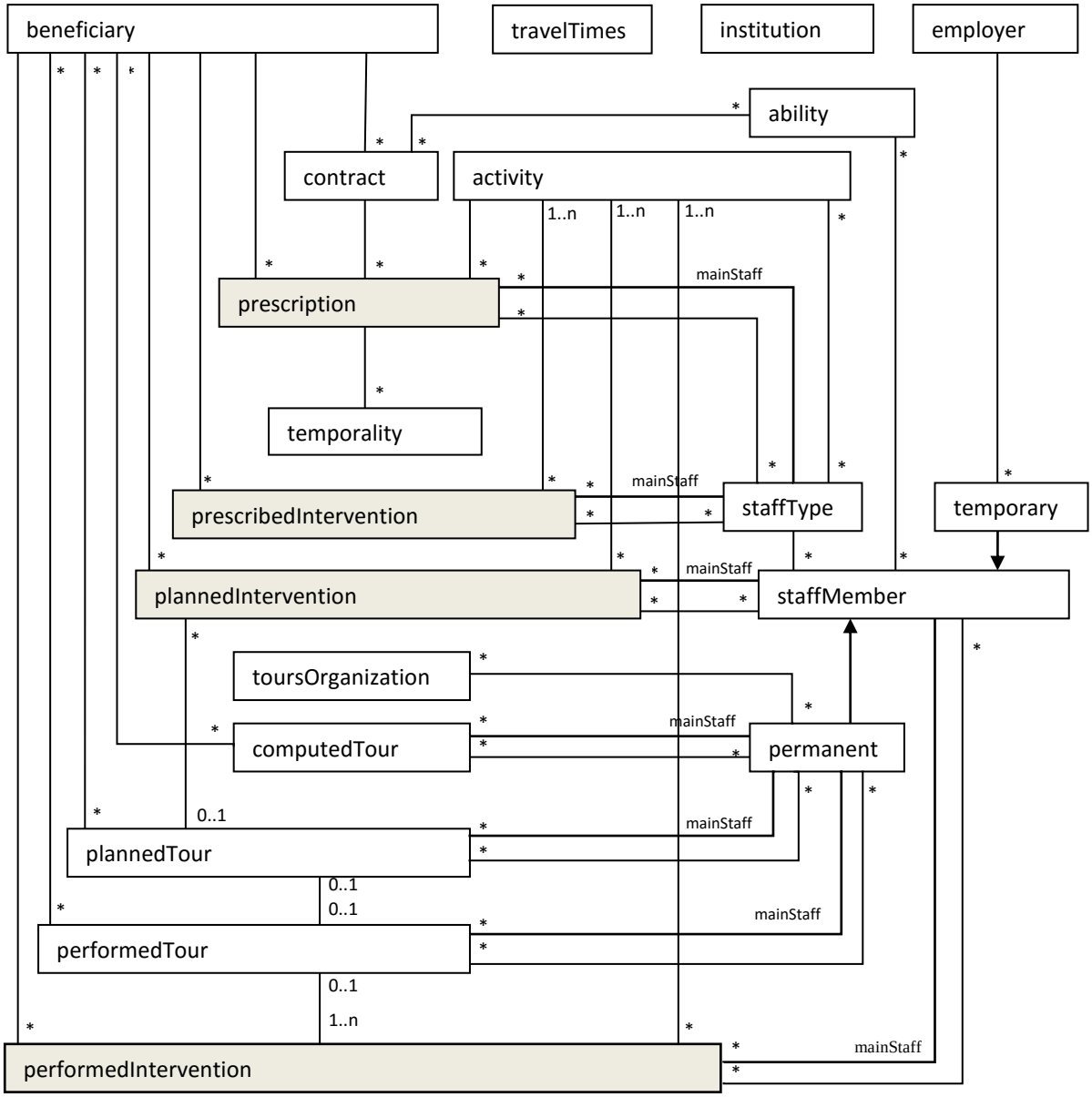

Figure 3: The conceptual data model
(Cardinalities of an association are placed to the right of vertical segments
at the to the top of horizontal segments; the default value is 1)

However, the report must include the start time of the intervention, the end time, the activities carried out, their durations and a way to include comments. If a wireless connection is available along the entire route of the tour, such input and output information can be transmitted in real time; otherwise, report information must be stored locally until the connectivity is restored. A tablet offers a comfortable form factor but is not essential. The experiments that were conducted as part of the Plas'O'Soins project [3] demonstrated that using smartphones was acceptable.

A related problem is managing delays in a tour. In some cases, it is important that the staff member notify the coordinator and the customer when a delay occurs so that readjustments can be made immediately. The use of smartphones is very useful for this purpose. Live information exchange also allows staff to react immediately in case of alerts. Storing such reports makes it possible to exploit the underlying information offline to conduct various types of analyses, for example, analysing the actual duration of completed actions with the goal of readjusting the estimated duration for each type of activity, analysing the interventions assigned to non-permanent staff to help with decisions on hiring permanent staff, analysing incidents or problems encountered during execution to adjust procedures, and so forth.

4.7 Conceptual Diagram

To synthesize this conceptual presentation, we propose the UML class diagram shown in Figure 3, which brings together the key concepts we have described and their associations. The prescription class in Figure 3 handles activities specification and is the focal point. Each instance of this class represents a specification in condensed form. The three views described earlier for interventions correspond to the three classes *prescribedIntervention*, *plannedIntervention* and *performedIntervention*. There is an instance of these classes for each day that involve the activities of an intervention. The organization and calculation of tours are made only for permanent staff members.

Prescriptions may be determined outside of any contract. Interventions can be planned and/or carried out without being prescribed (for example to cope with an unexpected situation). In terms of staff members, we distinguish the main member (who may be the only operator) and other staff members. The characteristics of the institution - and in particular its GPS coordinates - are stored in the institution class (one instance). The *travelTimes* class stores the travel time between any pair of destinations (including the institution). The institution is considered as point of arrival and departure for tours. Two directions of travel are distinguished because the travel times can be different. These travel times are used to calculate tours and are re-evaluated any time a change occurs in a customer's GPS coordinates (change of residence or new customer).

4.8 Simplified Versions

It is possible to envisage simplified versions of this conceptual model, for example, by assuming only one staff member is required per intervention and/or assuming that activity grouping is not useful. In such cases, calculating interventions and tours becomes considerably simpler. For example, simplifications such as these may be suitable for a machine maintenance company.

5 NoSQL Logical Models

This section describe our NoSQL logical models. We first present our data model in Section 5.1 and our services model in Section 5.2.

5.1 Data Model

The section 5.1.1 introduces the notations, which we use to describe our data model. The model itself is discussed in the section 5.1.2.

5.1.1 Notations

We base our framework on a "document" oriented model. Different NoSQL systems [12] are based on a representation of data by documents. We use the following notations based on those used in JSON [24]:

- Braces ("{" and "}") enclose the description of a document's or subdocument's structure.
- Brackets ("[" and "]") indicate repetition of a field or a sub-document.
- A colon (":") separates the name of a document or a subdocument from its description.
- Metadata and string values in a document are enclosed by double quotes; a colon (":") separates the metadata name from its value.
- The identifier for a document is appended by the metadata "_id" when a document's identifier is provided by the user; otherwise, the system generates an identifier automatically.

In addition, a document can reference the identifier of another document. Such references are expressed using the metadata "*nomDocument_id*". This type of reference corresponds to the concept of foreign key and is a simple way to represent a one-to-many association. A document-oriented model can support both

standardisation (synonymous with flexibility to changing data) and encapsulation (synonymous with limitation of joins). However, encapsulation can degrade performance when the database becomes very large, so it is necessary to maintain a wise balance between these two approaches [14].

Documents that represent instances of the same class are grouped into collections. The documents in a collection do not necessarily have the same format. A document-based model does not generally include schema. However, in our proposal, all documents that belong to the same collection have the same format. Below, we list the format corresponding to each collection.

5.1.2 Formats of the Documents

The logical model that stems from the conceptual model in Figure 3 consists of 15 document types (collections). Each class represents a document type—with some exceptions. For example, "temporality" is encapsulated in the document type "prescription", while "staffType" and "employer" are encapsulated in the document type "staff". The n-n associations all involve a class A that represents dated information and a class B which represents undated information. They are modelled by encapsulating references to the documents of the B collection in each document of the A collection. Below are the formats for the 15 document types along with some comments.

```
(D1)
beneficiary:
{
  "_ id ": string, "name": string,
  "firstName": string,
  "address":   string,
  "GPS":
  {
     "X": string,
     "Y": string
  }
}
```

The document "beneficiary" corresponds to a customer for whom a services contract is in place. The geographic location is based on the customer's GPS coordinates. These coordinates are used to calculate routes and tours.

```
(D2)
contract:
{
  "_ id ": string,
  "beneficiary_id": string,
   "startDate": date ,
   "endDate": date,
   "establishedOn": date,
   "establishedBy": string,
   "requiredAbilities":
       ["abilityName": string],
   "objectives": string,
   "characteristics": string,
   "contacts": string,
   "modalitiesRate": string
 }
```

The document "contract" specifies the services to be achieved. It specifies the particular skills required to fulfil the services given the current situation. The subdocuments "objectives", "characteristics" and "contacts" are specific to each application. For example, in a home care scenario, the subdocument "objectives" includes the main reason and any secondary reasons supplied for hospitalization. These reasons are duly listed by the medical authorities and institutions are obliged to comply. The subdocument "characteristics" includes the patient's physical characteristics (weight, height, etc.). The values of these characteristics can affect the treatment to be performed. Finally, the subdocument "contacts" notes people who can assist the beneficiaries, such as their doctor, pharmacist, nurse, social worker, husband or wife. In contrast, in the case of a cleaning company, the subdocument "objectives" would indicate the cleaning goals, and the subdocument "characteristics" would indicate the usual conditions of the premises and any difficulties that might be encountered.

```
(D3)
prescription:
{
   "beneficiary_id": string,
   "contract_id": string,
   "establishedOn": date,
   "establishedBy": string,
   "activityName": string,
   "activity Duration": integer,
   "temporalities":
       [{"period": string,
         "days": string,
         "timeSlots": string}],
   "mainStaffStatus": string,
   "mainStaffType ": string,
   "otherStaff":
       [{"staffStatus": string,
         "staffType": string}],
   "comment": string
}
```

The document "prescription" is used to declare an activity that is proposed to a beneficiary by the institution. This declaration is made in a condensed form through the temporal triplet (period, days,

intervals) presented in Section 4.2. These indications permit to generate the daily interventions to achieve for the beneficiary (document (D4)).The specified activity must be referenced in the document (D9) that also provides the standard duration of the activity, the types of staff able to handle it and the unit price.

```
(D4)
prescribedIntervention:
{
    "beneficiary_id": string,
    "date": date,
    "mainStaffStatus": string,
    "mainStaffType": string,
    "activities": [{
        "activityName": string,
           "activityDuration ": integer,
           "otherStaff":[{
              "StaffStatus": string,
                 "StaffType": string}],
    "timeIntervals": [{
        "arrivalTime": string,
        "departureTime": string}],
    "comment": string}
```

The document "prescribedIntervention" gathers all the prescribed activities that can be performed on a particular day in the premises of a beneficiary by the same type of staff member in a sequential manner (see the three principles outlined in Section 4.4). The intervention is expressed on each date of the period (as in the document "prescription", it is expressed in a condensed form). This grouping of activities is a preliminary stage for calculating the tours. The main objective is to minimize travel. The generation of interventions is a complex problem performed by a specialized service (see section 5.2.2).

```
(D5)
plannedIntervention:
{
   "beneficiary_id": string,
   "date": date,
   "mainStaffName": string,
   "arrivalTime": string,
   "departureTime": string,
   "plannedTourNumber": integer,
   "activities": [{
      "activityName": string,
      "activityDuration": integer,
      "otherStaff":
         [{"staffName": string}],
   "comment": string
}
```

Document (D5) is generated automatically for each intervention assigned to a planned tour. Interventions performed by temporary staff are not included in tours. For this type of intervention, the coordinator must create the corresponding document by providing the staff names and by assigning a 0 value for the tour number. Some home care institutions may systematically use temporary staff. In such cases, help can be envisaged for the creation of relevant documents. For example, one could imagine a service performing a solicitation by e-mail (on the basis of a list of contacts) and automatic document creation when a positive response is received.

```
(D6)
performedIntervention:
{
    "beneficiary_id": string,
    "date": date,
    "mainStaffName": string,
    "arrivalTime": string,
    "departureTime ": string,
    "plannedTourNumber": integer,
    "activities": [{
        "activityName": string,
        "activityDuration": integer,
        "otherStaff":
           [{"staffName": string}],
    "comment": string}}
```

The document (D6) is formatted the same as the previous one and initialized by the system using information resulting from (D5). The staff member completes and uploads it at the conclusion of the intervention.

```
(D7)
staff:
{
   "_id ": string,
   "name": string,
   "firstName": string,
   "address": string,
   "typeName": string,
   "statusName": string,
   "employer":
      {"employerName": string,
      "employerAddress": string},
   "ownedAbilities":
      ["abilityName": string]
}
```

The document (D7) provides information on the staff members who are qualified to perform the interventions. For each member, it stores status, type and that staff member's specific abilities. This latter information could be used to optimize the allocation of staff to patients and tours.

```
(D8)
institution:
{
   "institutionName": string,
   "address": string,
   "GPS": {"X": string, "Y": string}
}
```

The document (D8) is used to locate the institution geographically. It is assumed that the institution is the starting point and the ending point of each tour.

```
(D9)
activity:
{
   "activityName" : string,
   "activityDuration": integer,
   "activityCategory": string,
   "staffTypes": ["typeName": string],
   "priceUnit": string
}
```

The document "activity" lists the activities that the institution is able to implement. Each activity lists the types of staff required to achieve it. For example, for home care, a toilet activity may be performed by a either a nursing assistant or a nurse, but an injection may only be performed by a nurse.

```
(D10)
ability:
{
   "abilityName": string,
   "comment": string
}
```

The document (D10) lists specific abilities of the institution's staff. For example, for home care, an ability could be "Alzheimer's" or "end of life". These abilities are independent of the staff type (nursing assistant, nurse) or status (permanent, temporary). Listing these abilities is important for assigning staff members to tours. For example, a nurse able to handle Alzheimer's disease should be preferentially assigned to a tour that includes an "Alzheimer's" patient.

```
(D11)
toursOrganization:
{
   "date": date,
   "timeIntervals": [{
      "startTime": string,
      "endTime": string,
      "staffType": string,
      "numberOfTours": integer,
      "staffs": ["staff_id": string]}]
}
```

The document (D11) allows the institution to choose the organization of its tours. A tour occurs at a given date in a given time interval (morning, 15-18h ...), with a given type of staff. Tours are grouped by interval and type of staff. The number of staff members available in an interval must be at least equal to the number of tours for this interval.

```
(D12)
computedTour:
{
   "date": date,
   "computedTourNumber": integer,
   "staff_id": string,
   "startTime": string,
   "endTime": string,
   "visits": [{
      "beneficiary_id": string,
      "arrivalTime": string,
      "departureTime": string,
      "otherStaff":
         ["staff_id": string]}]
}
```

The document (D12) stores results provided by the planning service (see section 5.2.2). The interventions to be performed on a given date are distributed among the various tours declared in the document (D11). A staff member is assigned to each tour. The arrival and departure times at each home are calculated by taking into account the intervention durations and the travel times.

```
(D13)
plannedTour:
{
   "date": date,
   "plannedTourNumber": integer,
   "staff_id": string,
```

```
   "startTime": string,
   "endTime": string,
   "visits": [{
      "beneficiary_id": string,
      "arrivalTime": string,
      "departureTime": string,
      "otherStaff":
         ["staff_id": string]}],
   "comment": string
}
```

The automated service dedicated to planning and touring does not always provide a result that completely meets the coordinator's needs. Manual adjustments are sometimes necessary to take constraints or unexpected events that occur at the last moment into account. The document (D13) records the planning the coordinator really wants. From the "computedTour" document, the coordinator can modify or delete a tour, or create a new tour.

```
(D14)
performedTour:
{
   "date": date,
   "plannedTourNumber": integer,
   "staff_id": string,
   "startTime": string,
   "endTime": string,
   "visits": [{
      "beneficiary_id": string,
      "arrivalTime": string,
      "departureTime": string,
      "otherStaff": ["staff_id": string],
      "comment": string}]
}
```

The document (D14) records the characteristics of completed tours. These characteristics are captured by a staff member assigned to the tour. They can be entered into a fixed computer at the end of the tour, or (better) from a mobile device (smartphone or tablet) at the end of each visit.

```
(D15)
travelTimes:
{
   "between": [{
         "beneficiary_id1": string,
         "beneficiary_id2": string,
         "time": integer}],
   fromInstitution: [{
         "beneficiary_id": string,
         "time": integer}],
      toInstitution: [{
         "beneficiary_id": string,
         "time": integer}]
}
```

The document (D15) records the travel times between each pair of homes and between each home and the institution. We consider the two directions for each route since the travel times are not necessarily the same.

5.2 Services Model

The processing to be performed as part of process P2 (activities management) can be grouped into five broad categories: CRUD services, computing services, display, analysis and archive services.

5.2.1 CRUD Services

The CRUD services (Create, Read, Update and Delete) are low-level services that enable document management. An overview of the different services which are needed is given in Table 3.

C and R services are valid for all documents. Documents (D4), (D5) and (D12) are created automatically by the system. All others documents must be created by users. U and D services are not available for all documents. For (D1), (D2), (D7), (D8), (D9), (D10) and (D11), U is allowed but D is prohibited. These documents serve as references and uncontrolled removal could cause integrity problems. Moreover, it is useful to keep them to allow analysis on historical data.

Update is valid for (D3)—but only to stop a prescription at a given date. Delete is allowed for (D4), (D5), (D12) and (D13) as a result of updates to documents (D2) and (D11). For example, when a prescription is stopped at a date d, it is indeed necessary to recalculate the interventions and thus the tours that had already been planned from date d + 1. Similarly, we must recalculate tours if document (D11) is the subject of an update or a delete. These deletions are automatically ensured by two specific services: cascadeUfromD2 and cascadeUDfromD9.

Documents (D6) and (D14) are created from information provided by a staff member after completing an intervention and are never updated or deleted. They are kept exactly as created to support activity tracing and historical analyses. C and U services are activated through specific interfaces that require the user to introduce valid values for foreign keys. Such CRUD services guarantee the referential integrity of the data.

Table 3: Synopsis of CRUD services for 15 documents

Documents	Create	Read	Update	Delete	Delete from
(D1) beneficiary	x	x	x		
(D2) contract	x	x	x		
(D3) prescription	x	x	x (stopped)		
(D4) prescribedIntervention	x (A)	x			x (from update (D3))
(D5) plannedIntervention	x (A)	x			x (from update (D3))
(D6) performedIntervention	x	x			
(D7) staff	x	x	x		
(D8) institution	x	x	x		
(D9) activity	x	x	x		
(D10) ability	x	x	x		
(D11) toursOrganization	x	x	x	x	
(D12) computedTour	x (A)	x			x (from update (D3)) x (from update or delete (D11))
(D13) plannedTour	x	x			x (from update (D3)) x (from update or delete (D11))
(D14) performedTour	x	x			
(D15) routesTime	x(A)	x	x(A)		

```
Enter the date
Read the travel times document
For each interval in the date / *these intervals are available in the document toursOrganization */
        For each type of staff
            Read the number of tours
            Read interventions for the date for the interval and for type of staff with status = "Permanent"
            Call the solver to compute the tours in the interval
            If the solver has found a solution within a delay TMax
                Then store each computed tour in a computedTour document
                Otherwise display a "automated planning not possible - perform manually" message
            EndIf
        EndFor
EndFor
```

Listing 4: Scenario for implementing the computeTours service

Table 4: Useful services

Name	Purpose	Used documents
Computing Services		
computeInterventions(date)	Group the activities of each beneficiary	(D3), (D4)
computeTours(date)	Compute the tours to optimize a global criteria	(D5),(D11), (D12)
computeTravelTimes()	Compute travel times	(D1), (D15)
Display Services (examples)		
displayPlannedTours (date, staff_name)	Display the planned tours for a given staff with the name of the beneficiaries, their addresses and the visit schedules.	(D1), (D5), (D13)
displayPlannedInterventions (period, beneficiary_name)	Display the planned interventions for a given beneficiary with the staff name, the activities and the visit schedules.	(D1), (D5), (D13)
displayPerformedInterventions (period, beneficiary_name)	Display the performed interventions for a given beneficiary with the staff name, the activities and the visit schedules.	(D1), (D6), (D14)
Data Analysis Services (examples)		
averageActivityDuration (activity, period)	Compute the average value for the duration of the activity observed over the given period	(D6)
sumHours(period, staff_name)	Compute the total duration of the interventions performed by a given staff for the given period	(D6), (D7)

5.2.2 Computing Services

Two specific services are devoted to generating interventions and planning (Table 4). These two services are linked because the generation of interventions is a preparatory step to planning. The generation of interventions complies with the conditions defined in Section 4.4. There may be conflicts between two interventions when they occur in overlapping time windows with two different staff members. The service should detect conflicts. When a conflict is detected, the system must send a message to the coordinator to notify him to change the time specifications for the activities involved.

Planning is complicated because of the variety of constraints to be considered (see section 4.5). Another difficulty is the systematic capture under operational conditions of all the data required for accurate calculation. It is important to consider automatic planning as an aide because that view means the problem the system solves is to create results in a form suitable for a coordinator to make modifications easily. It is then important to indicate to the user any changes that would violate a strong constraint.

Our framework incorporates documents that express the most important constraints: respecting the type and status of a staff member, respecting the time intervals for an activity and matching staff abilities with the abilities required for a patient. The system can easily be extended to record other types of data necessary to express additional constraints, for example, respecting the patient preferences for certain staff members or respecting a patient's time constraints. An important contribution of this framework is facilitating communication between the data preparation services and the planning service. Listing 4 shows a scenario for the organization of this service. The computing of tours can be performed by a solver or by a specific algorithm. It can also be performed manually if the number of tours and interventions is not too large.

A third service calculates travel times (Table 4). Different approaches can be used to implement the service, for example, it could make an initial estimation and then subsequently readjust it by taking into account the observed time during the first few visits or use road mapping software based on addresses or use software such as Google Maps, which calculates travel time based on GPS positions.

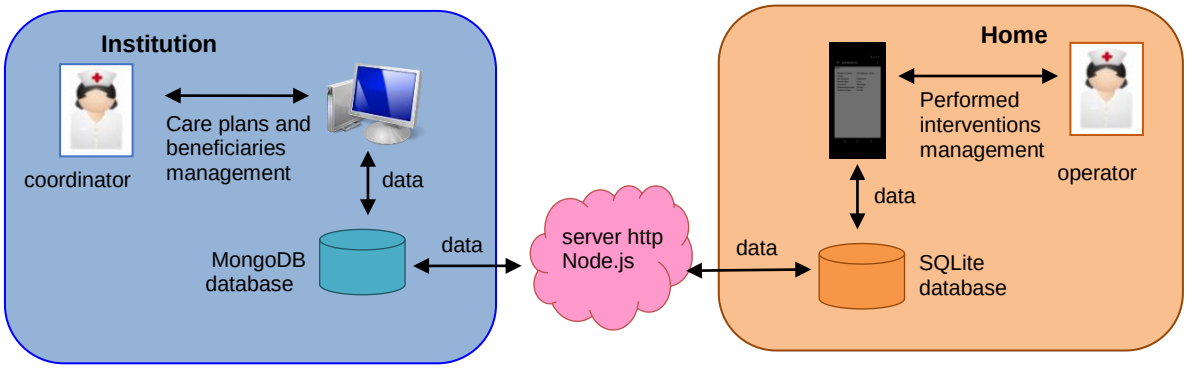

Figure 4: Architecture for implementation

5.2.3 Display Services

Most helpful displays can be obtained from the R services of documents. For example, the tours to be conducted by a staff member on a given day will be retrieved by the R service (D13). However, this display requires a user to enter the *staff_id* and provides only the *beneficiary_id* and visiting hours. To obtain a display which lists the activities to be performed, the names of beneficiaries and their addresses from the staff name, we must build a more elaborate service that joins documents (D1) and (D4). Table 4 illustrates examples of display services involving several documents.

5.2.4 Analysis Services

To improve the management of the institution, it is important to compute a number of indicators for a given past period. Therefore, data must be kept for a sufficiently long period. Table 4 provides examples of services for computing indicators.

5.2.5 Archiving Services

It is not possible to store all the data locally over a long period. We therefore suggest the use of a repository that can be accessed on demand. We can see that all documents, except (D1), (D7), (D8), (D9) and (D10), incorporate a usage date. Therefore, for example, we can envisage archiving regularly (such as every 2 years) any documents whose usage date is more than four years from the current date. This archival process must maintain the integrity of embedded references to enable any type of research on past data.

6 Implementation with MongoDB

We propose to implement our framework by using the MongoDB system [2]. Our goal is to identify the possible facilities a NoSQL system can bring. For this test implementation, we do not use documents (D2) and (D10) because we do not consider the abilities of staff members when planning. To create the tests we developed a data set relating to home care.

6.1 Architecture and Conditions of Implementation

Browser-based interfaces concerning prescribed and planned interventions are activated from a terminal connected to a Windows 7 server. The server hosts the database. We used version 2.6 of MongoDB which can be downloaded for free. A smartphone running Android is made available to each staff member in charge of interventions. Interfaces concerning performed interventions are activated from the smartphones. At the beginning of a tour, the staff member initiates a web exchange with the server to load the characteristics of the interventions to be performed, and, at the end of the tour, the staff member launches a new web exchange to transmit the updated values (Figure 4).

Data stored on the smartphones is managed with the SQLite database system. We use this system because SQLite is integrated with Android (version 4.4.2) and therefore requires no special software installation. The applications on the server and on the smartphones are coded in the Java language. Data exchange between MongoDB and SQLite is performed via the server using Node.js (version 0.10), which is a platform that enables the use of JavaScript on both the server and client side.

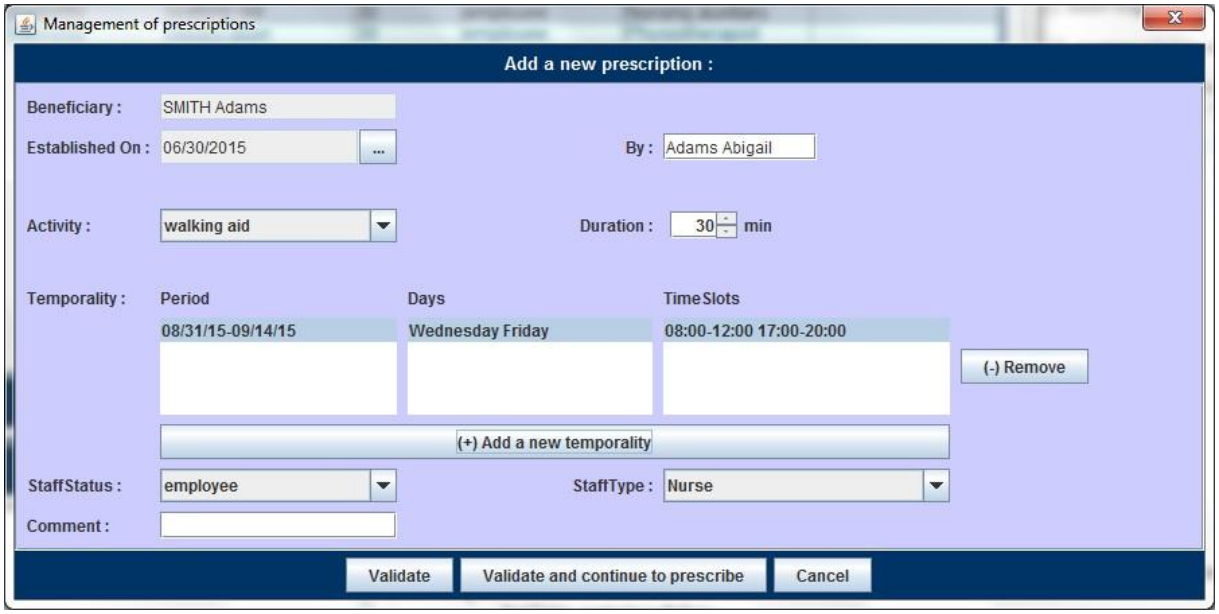

Figure 5: Interface for creating a new prescription

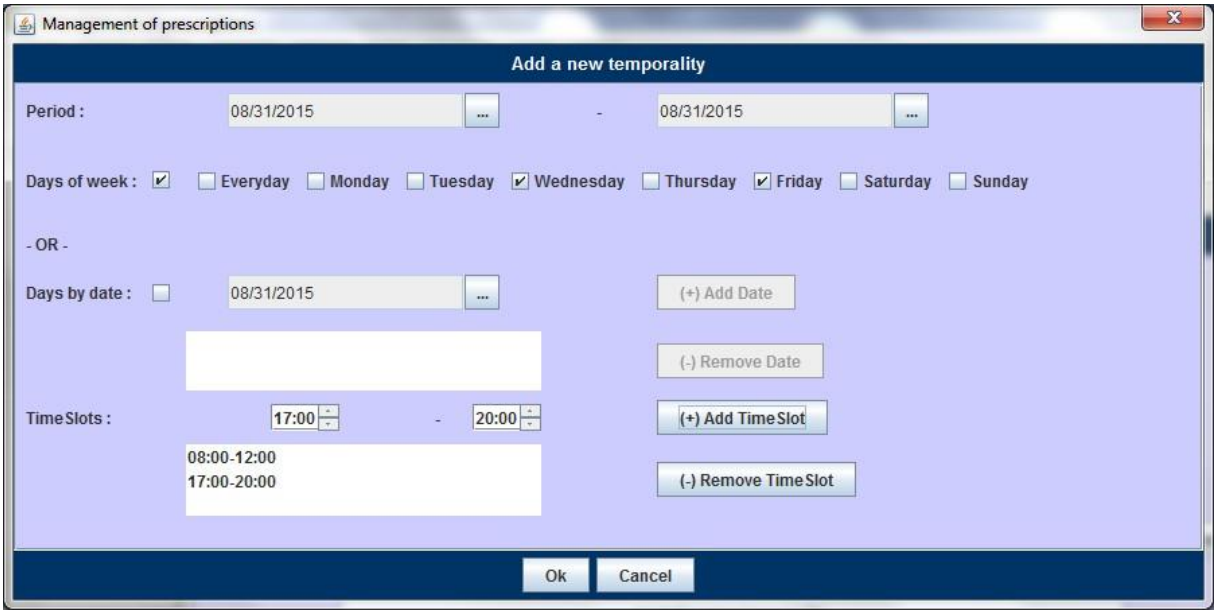

Figure 6: Interface for creating the temporality of a prescription

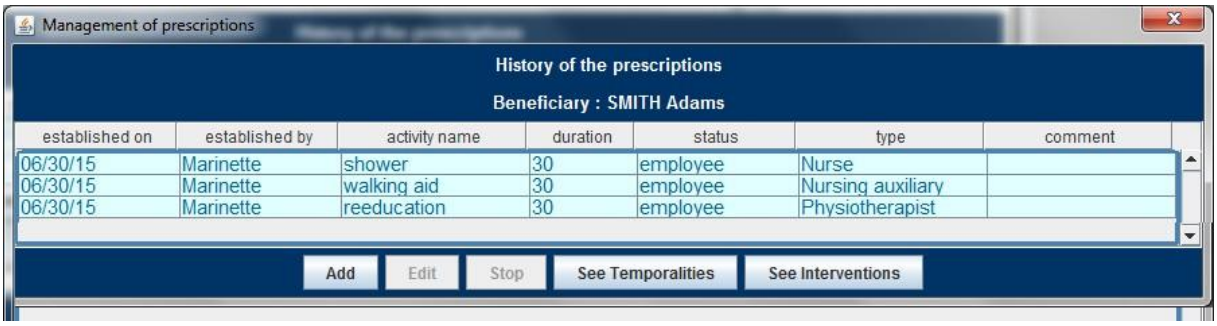

Figure 7: Interface for displaying the prescriptions of a beneficiary

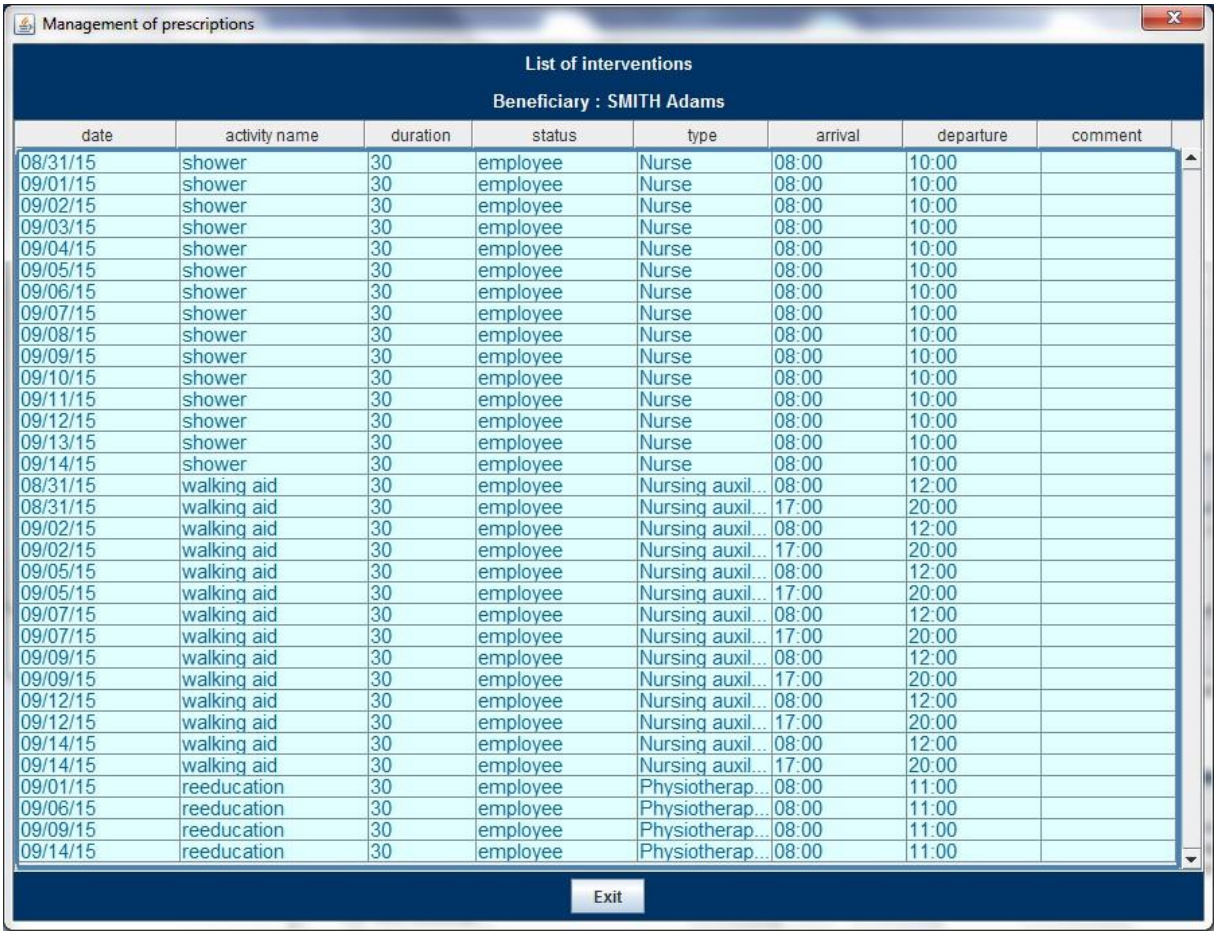

Figure 8: Interface for displaying the prescribed interventions

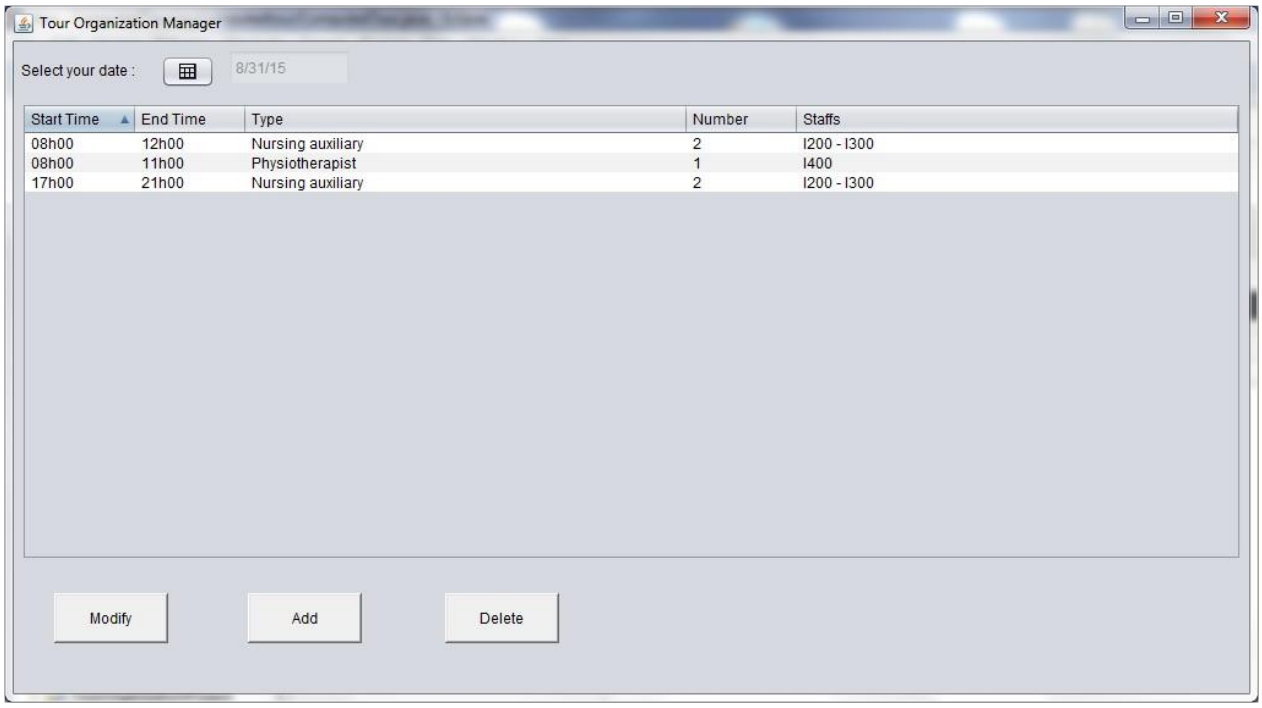

Figure 9: Interface for specifying tours organization

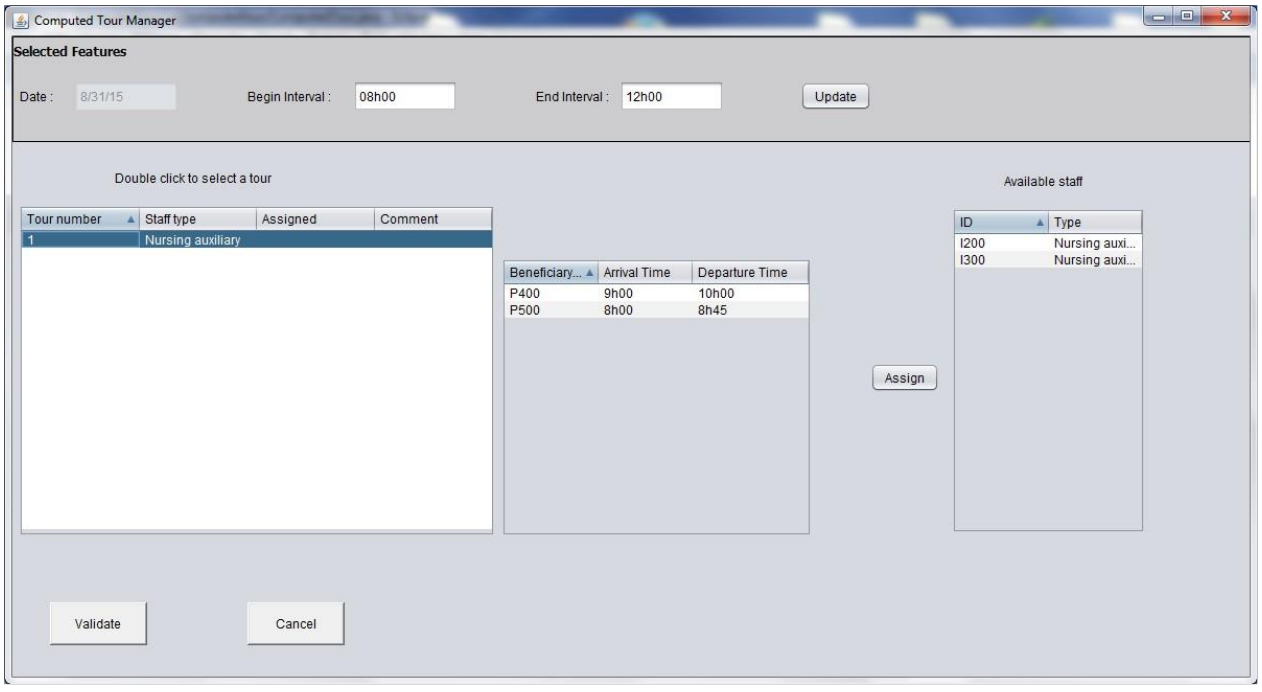

Figure 10: Interface for allocating staff to computed tours

Three groups of computer science students programmed the system as part of their project work. The effort involved was distributed by views: a one member group was responsible for the prescribed view (documents (D1), (D3), (D4) and (D9)), a pair for the planned view (documents (D5), (D7), (D8), (D11), (D12), (D13) and (D15)) and another pair for performed view (documents (D6) and (D14)). Each group created services related to the documents it was responsible for. Each group worked over a 4-month period at the rate of fifteen hours per month. These students did not know MongoDB and Android at the outset, and this project represented an opportunity for them to learn these two systems.

6.2 Development

First, we established a representative data set to populate the database. Each group initialized its work from the following items: the data set, the format of each document type and the service descriptions. The data set includes 10 beneficiaries whose homes are located within a 30 km square urban area. The care institution is located north of that area. Travel times were assessed from aerial distances on the basis of an average travel speed of 50 km/h. For each beneficiary, we considered two to three prescriptions. For example, here are the care prescriptions for the patient SMITH Adams from 08/31/2015 to 09/14/2015:

- shower - every day - morning
- walking assistance - Monday, Wednesday and Friday - morning, evening,
- rehabilitation by a physiotherapist - Tuesday, Thursday – morning

Interfaces were developed in Java from the descriptions of documents and services and designed taking into account the feedback we experienced from the Plas'O'Soins project [3]. We provide some examples of these interfaces below. Figure 5 shows the interface for creating a prescribed activity. We can make changes as long as the data have not been recorded in the database. Temporalities are defined via the interface of Figure 6. Remember that it is possible to define several temporal expressions for an activity.

When a prescription is validated, a new document (D3) (prescription) is created in the database. It is possible at any time to display a beneficiary's prescriptions using the interface shown in Figure 7. This display corresponds to the notion of a "care plan" as is typically defined in hospitalization services. The "Stop" button stops a prescription. The "Edit" button allows a user to edit a prescription. When a prescription is edited, the current prescription is stopped and a new one is started. Figure 8 shows the interface for displaying daily planned interventions for a beneficiary over the next two weeks. This display is obtained by selecting the *computeInterventions(d)* service for each day d of the period.

The interface shown in Figure 9 allows the coordinator to specify the organization of tours for a given date. The coordinator must allocate available staff members to each tour. This information is used directly by the *computeTours()* service to calculate tours. Given the small size of the collections in this study, we performed this calculation manually. For larger data sets, the application might call a solver such as GLPK via the appropriate dll (java/glpk in this case) to offer a pre-planning coordinator. For each computed tour, the coordinator must then designate one of the available staff members as the main staff member by using the interface in Figure 10. This interface displays the visits that compose the computed tour. After a staff member has been assigned, that person's name is removed from the list of available staff.

The interfaces of Figure 11 show how a staff member can view planned tours for a given day and update a planned intervention to generate the intervention that will actually be performed. In the example shown, the staff could not begin the operation at the scheduled time. This intervention was composed of two activities and the staff member has performed only the first activity.

6.4 Personalization

Our framework can be customized to better meet the needs of any particular type of service agency. An example is shown below where we highlight the facilities offered by MongoDB for this type of modification. The framework must be arranged to enable the coordinator to assign a priority to each prescription. Documents (D3), (D4) and (D5) and the *computeTours(date)* service are impacted by this modification. Document (D3) is entered by the coordinator, so it requires adding a new "priority" field to the input screen. Documents (D4) and (D5) are automatically generated by the system, so we have just to change their writing in the database by adding the new field. The *computeTours(date)* service must be adjusted to take the priorities into account (interventions with higher priority activities should be placed first in a tour). A SQL implementation of this framework would have required more modifications because it would also be necessary to modify the structure of the relational tables corresponding to documents (D3), (D4) and (D5). It appears that a NoSQL approach facilitates adaptations because the number of elements to modify, include or delete is less important than with an SQL approach. In addition,

metadata embedded in MongoDB facilitate the identification of elements.

6.5 Generalization

To show how this framework can be generalized to other domains, consider an example for a business that cleans a set of offices occupied by the AIRASEC society. For this contract, the prescriptions from 08/31/2015 to 09/14/2015 are:

- wash floors - every day - morning - 30 minutes - two first grade member staff.
- clean writing desk - Monday, Wednesday and Friday - morning - 20 minutes - one first grade member staff.
- polish floors - Monday - evening - 80 minutes – one second grade member staff.

These prescriptions can be specified easily using our temporal language. The *computeInterventions(date)* service groups the first two activities on Mondays, Wednesdays and Fridays. For other days, the morning intervention includes only the single "wash floors" activity. Finally on Monday evening an intervention will be made separately to accommodate the "polish floors" activity. Staff allocations and tours will be calculated by the *computeTours(date)* service by considering interventions for all customers.

More generally, our framework is capable of processing all services whose activities may be described with our specification language, i.e., activities that occur during intervals in a day. Moreover, the language can be easily extended to allow the specification of activities which repeat over weeks or months and for which the intervals of realization cover several days. Appendix A provides an overview of such extension which would make the system suitable for application in many domains and especially construction and civil engineering.

6.6 Balance Sheet

Throughout this implementation, the objective was to verify the conditions for implementing our framework with a NoSQL system. Compared to relational systems, NoSQL systems have advantages - but also drawbacks - that depend on the type of application. It is therefore important to evaluate whether the benefits are truly significant for our framework. In Table 5, we considered various criteria and have supplied our evaluation of each. Gaining an understanding of the logical model and subsequently implementing the physical model by the three groups of developers was quite rapid. Cooperative development posed no difficulty because the distribution of work across the three views was simple and natural.

Document sharing was not difficult because each document has a specific format that must be respected. We did face integrity problems with the description of services. Referential integrity comes from the references between documents. These references have been reduced to a minimum, and the system performs referential integrity checks when the interfaces are activated. We also wanted transactional atomicity capability to guarantee that updates to (D3) and (D1) would be propagated to several other collections. However, transactional operations are not supported by MongoDB. Therefore, we must implement it into the application. We have not tested any competitive solutions, but our framework does not require high performance at this point.

Competition arises mainly when several staff members access the server at the same time to obtain information about their tours or to update them. Documents in MongoDB are locked through collections. In the absence of an active update operation, multiple readings may occur simultaneously from a collection; however, when a write lock has been set on a document, that lock prohibits write operations to any document in the collection. Therefore, write operations for documents in collection (D6) for interventions updates must be conducted in a strict sequence, which may cause delays. However, these delays can be managed transparently at the smartphone level for staff members.

The developer groups could not devote much time to finalizing screens, and ergonomic improvements are needed. Using interface-editing software would probably provide more efficient development at this level. Despite the absence of transaction management, we believe that MongoDB is a good candidate for implementing our framework. A framework must be enriched and personalized, and on these points, MongoDB is effective.

7 CONCLUSIONS AND PERSPECTIVES

Summary of our proposals:

Various commercially available software tools have been designed to help service providers organize their activities in private homes or in local businesses; however, these tools are relatively expensive and are out of the reach of smaller companies. No current unified approach exists to allow economies of scale. Observing that mobilized concepts and operations do not differ greatly from one domain to the other, our goal in this paper was to propose a generic framework for the field of geo-localized services.

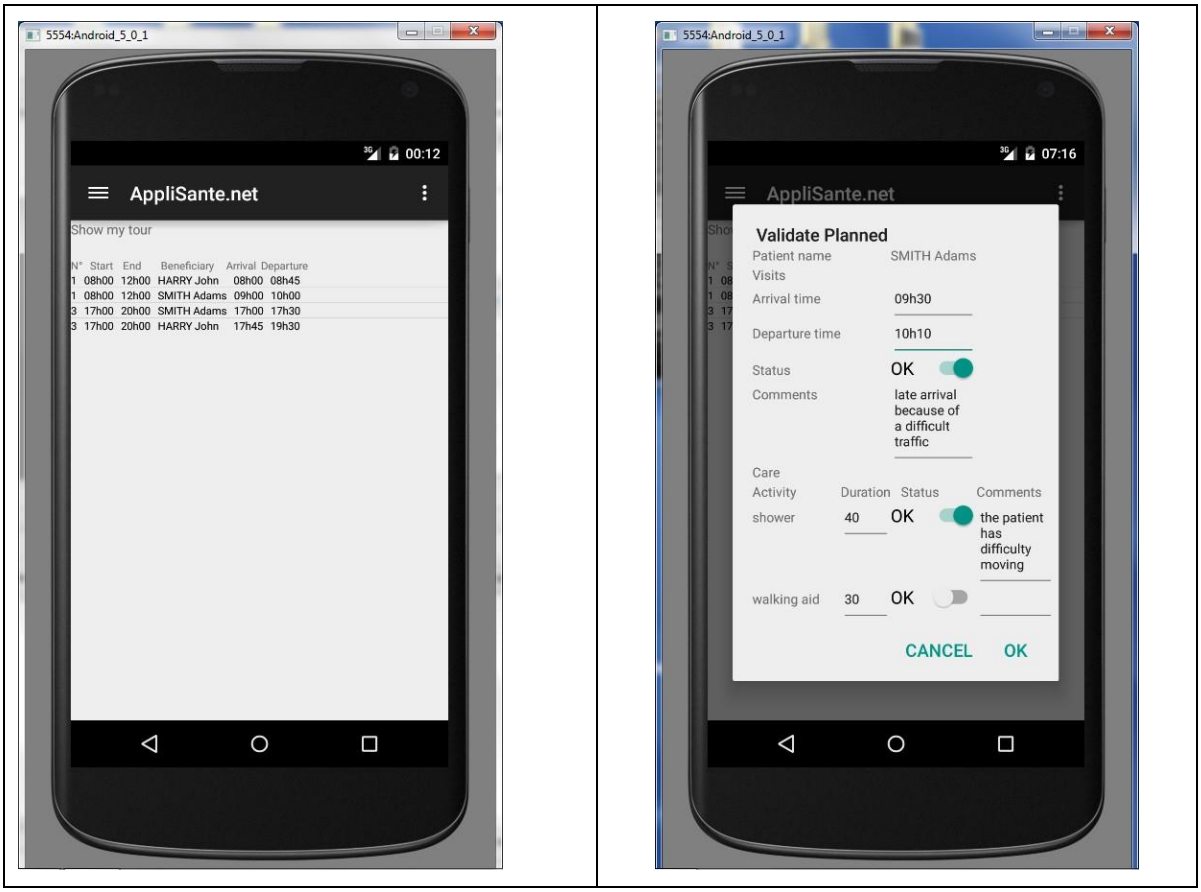

Figure 11: Smartphone interfaces for reading and updating interventions

Table 5: Criteria evaluated through the implementation

Criteria	Our evaluation
Quality of logical oriented document modelling	Good
Acquire developers understanding of the logical model	Easy
Development of physical model by developers	Easy
Cooperative development effectiveness	High
Adequacy of the physical model with interfaces	Good
Flexibility	High
Referential integrity	Good
Handling complex queries	Fair
Concurrency	Not evaluated but not critical
Transactional management	Poor

This framework provides a unified description of processes. The core process supports the management of activities. It is subdivided into three sub-processes that correspond to the three views of activities: prescribed view, planned view and performed view. These views are based on an original temporal model that supports descriptions of recurring activities for each customer. These activity descriptions may be created at different levels of granularity and can adapt well to a variety of real-life situations. The framework assembles all the information required for planning and routing and supports both manually or automatic achievement of those tasks. The proposed concepts were illustrated using two different areas: home healthcare and on-premises cleaning services. This analysis resulted in a generic conceptual model that can be implemented using either relational systems or NoSQL systems. Our proposed model can be considered as a domain-specific language (DSL) for the service sector.

We then conducted an implementation with a NoSQL approach in two phases: i) logical modelling based on "documents" and "services", and ii) implementation of the logical model using the MongoDB system. The logical model is simple and compact. The organization of treatments as services operating on documents is natural. Thus, the framework is easy to understand. We verified the ability of MongoDB to support rapid implementation thanks to the following facilities: convenience of physical implementation, ease of cooperation between developer groups and ease of evolution. We considered the well-known disadvantages of the NoSQL approach concerning referential integrity, concurrency and transactional management. In our framework, referential integrity and concurrency do not pose difficult problems; however, the absence of a transaction manager is a serious drawback.

A framework must also offer personalization capabilities. Thanks to its flexibility—and despite its drawbacks—we believe that a system such as MongoDB is a good candidate for the implementation of an adaptable and extensible framework. Through this prototype, we wanted to show the feasibility of the framework. We believe that our technical choices can be adapted to develop an industrial version.

Advantages of our framework:

This framework provides several important advantages over existing solutions:

1) It allows a significant reduction of development costs because an implementation based on a NoSQL DBMS is easier and faster than one based on a relational DBMS, making it possible to offer business solutions that are within the reach of small service agencies.

2) It is generic and can adapt to the needs of several domains, thus taking advantage of economies of scale.

3) Its NoSQL implementation is flexible enough to support specific adaptations for each type of service agency.

4) It structures the activities management process into three phases that correspond to separate modules. In particular, the data required for planning are organized in a clear and compact way. It is therefore possible to modify scheduling algorithms without redesigning the data structure.

5) The language proposed for specifying activities is new and is both structured and rich. By allowing specifications at several levels of granularity, it offers flexibility to managers in establishing activity plans.

SQL versus NoSQL approaches:

The commercial solutions are usually based on relational DBMSs. These systems have demonstrated their effectiveness and hold decisive advantages regarding integrity checking. NoSQL systems have gaps in integrity checking, but are easier to implement. Their data model is generally simpler and more compact. Thus, our conceptual model translates into 15 MongoDB document types, while an equivalent relational implementation would require more than 30 tables. In addition, structural changes are easier to manage using a NoSQL DBMS. It is necessary to study the advantages and disadvantages of each type of DBMS with respect to the intended application. In our case, it appears that the final balance is in favour of an implementation based on a NoSQL DBMS.

Perspectives:

The framework applies only to the activities management process and would need to be extended to incorporate data and services to support the other processes shown in Figure 1. The modelling of these other processes does not pose any particular difficulties, and we fully expect MongoDB to permit easy and coherent extensions to support them. It would be interesting to study how to deliver certain required equipment and consumables to homes by including them in the tour planning process. One solution would be to consider a delivery as an activity. Thus, a delivery could be grouped with another activity.

One point we consider significant is that of planning. It is very difficult to model a priori all the constraints that managers may face. Some constraints may occasionally appear at the last moment. We stressed the need to consider this framework a move

toward creating of a form of aid that permits the coordinator to finalize the result. A solution that allowed a coordinator to express constraints in a high level language would be most effective. The integration of such a language would probably require important restructuring of the planning data. We leave this important feature for future work.

REFERENCES

[1] Arcan, "AtHome software". http://www.arcan.fr/logiciel-suivi-soins-a-domicile.html, accessed April 1, 2015.

[2] R. Arora, and R. R. Aggarwal, "Modeling and Querying Data in MongoDB", *International Journal of Scientific and Engineering Research*, Vol 4, Issue 7, 2013.

[3] R. Bastide, P. Bardy, B. Borrel, C. Boszodi, M. Bouet, K. Gani, E. Gayraud, D. Gourc, E. Lamine, P.H. Manenq, M. Schneider, F. Toumani, "Plas'O'Soins: A software platform for modeling, planning and monitoring homecare activities", *Innovation and Research in BioMedical engineering (IRBM)*, 35(2), pp. 82-87, 2014.

[4] T. Bektas, "The multiple traveling salesman problem: an overview of formulations and solution procedures", *Omega*, Volume 34, Issue 3, pp. 209–219, 2006.

[5] B. Bachouch, "An optimization model for task assignment in home health care", *IEEE Workshop on Health Care Management (WHCM10)*, Venise, Italy, Feb. 2010.

[6] N. Bricon-Souf, F. Anceaux, N. Bennani, E. Dufresne, L. Watbled, "A distributed coordination platform for home care: analysis, framework and prototype", *International Journal of Medical Informatics*, 74(10), pp. 809-825, 2005.

[7] M. Bouet, K. Gani, M. Schneider, F. Toumani, "A general model for specifying near periodic recurrent activities - application to home care activities", *IEEE 15th International Conference on e-Health Networking, Applications & Services (Healthcom)*, pp. 207-211, 2013.

[8] M. Bouet, K. Gani, M. Schneider, F. Toumani, "Définition d'un DSL pour les soins à domicile", Livrable du projet Plas'O'Soins, Décembre 2014.

[9] Capterra, "Top Maid Service Software" http://www.capterra.com/maid-service-software/, accessed April 1, 2015.

[10] Consumeraffairs, "Compare Reviews for Best Home Health Care Software", http://www.consumeraffairs.com/online/home-health-care-software, accessed April 1, 2015.

[11] D. Gourc, F. Marmier, P. Gaborit, "An approach based on tabu search technique for solving a multi time window home healthcare scheduling", *10ème Conférence Francophone de Modélisation, Optimisation et Simulation*, 2014

[12] J. Han, E. Haihong, D. Le, J. Du, "Survey on NoSQL database", *6th International Conference on Pervasive computing and applications (ICPCA)*, 2011.

[13] J. Jemai, M. Chaieb, K. Mellouli, "The home care scheduling problem: a modeling and solving issue", *5th International IEEE Conference on Modeling, Simulation and Applied Optimization (ICMSAO)*, pp. 1-6, 2013.

[14] A. Kanade, A. Gopal, S. Kanade, "A study of normalization and embedding in MongoDB", *Advance Computing Conference (IACC), IEEE International Conference*, pp. 416-421, 2014.

[15] Y. Kergosien, C. Lenté, J. C. Billaut, "Home health care problem: An extended multiple traveling salesman problem", *4th Multidisciplinary International Scheduling Conference: Theory and Applications*, 2009.

[16] A. Király, J. Abonyi, "A novel approach to solve multiple traveling salesmen problem by genetic algorithm", *Computational Intelligence in Engineering*, Springer, 2010.

[17] V. G. Koutkias, I. Chouvarda, N. Maglaveras, "A multiagent system enhancing home-care health services for chronic disease management", *IEEE Transactions on Information Technology in Biomedicine*, 9(4), pp. 528-537, 2005.

[18] E. Lamine, A. R. H. Tawil, R. Bastide, H. Pingaud, "An Ontology-Driven Approach for the Management of Home Healthcare Process", *Enterprise Interoperability*, VI, Springer, pp. 151-161, 2014.

[19] E. Lanzarone, A. Matta, "Robust nurse-to-patient assignment in home care services to minimize overtimes under continuity of care", *Oper. Res. Heal. Care*, 2014

[20] W. Liu, S. Li, F. Zhao, A. Zheng, "An ant colony optimization algorithm for the Multiple Traveling Salesmen Problem", *4th IEEE Conference on Industrial Electronics and Applications*, 2009.

[21] A. Matta, S. Chahed, E. Sahin, Y. Dallery, "Modelling home care organisations from an operation management perspective", *Flexible Services and Manufacturing Journal*, 26(3), pp. 295-319, 2014.

[22] Medical Link, "MedLink product", http://www.med-link.org/, accessed April 1, 2015.

[23] O. Rejeb, R. Bastide, E. Lamine, F. Marmier, H. Pingaud, "A model driven engineering approach for business continuity management in e-Health systems", *6th IEEE International Conference on Digital Ecosystems Technologies (DEST)*, pp. 1-7, 2012.

[24] C. Severance, "Discovering JavaScript object notation", *Computer*, Volume 45, Issue 4, pp. 6-8, 2012.

[25] A. F. Torres-Ramos, E. H. Alfonso-Lizarazo, L. S. Reyes-Rubiano, C. L. Quintero-Araújo, "Mathematical Model for the Home Health Care Routing and Scheduling Problem with Multiple Treatments and Time Windows", *Proceedings of the 1st International Conference on Mathematical Methods & Computational Techniques in Science & Engineering*, pp. 140-145, 2014.

[26] D. V. I. Weppenaar, H. J. Vermaak, "Solving planning problems with Drools Planner a tutorial", *Interim: Interdisciplinary Journal*, 10(1), pp. 91-109, 2011.

APPENDIX A: FORMAL DEFINITION OF THE TEMPORAL LANGUAGE

For this formal definition, we use a BNF notation. We identify the repetition of an element by framing it into braces and by post fixing it with one of three symbols : "?", "*", "+" which respectively means that the element may be present or absent, is present any number of times, is present at least once.

Important: To simplify the writing of a list of contiguous successive elements, we use a shorthand notation by indicating only the first item in the list and the last one separated by "-" (e.g. Monday-Friday to list all weekdays from Monday to Friday). A simple temporal expression (see Listing A-1) is built from the triplet (period, days, intervals). A general temporal expression is as a combination of simple temporal expressions. A general expression is correct if the set of days coming for its simple expressions have an empty intersection. We define below each triplet (time, day intervals).

The form 1 for days is used to specify dates in the period (holiday is a shorthand to represent all dates of public holidays). The form 2 is used to specify weekdays. The form 3 specifies that the activity occurs repeatedly every day or every n days from a given date. The "except" keyword allows to exclude dates or days and then introduces irregularities.

Ex1: every(2) days(01/12/13) except(holiday) means every two days from 01/12/13 except holiday

Ex2: everyday except(Sunday) is equivalent to Monday-Saturday

The intervals are marked in a day by standard day parts (e.g. morning, afternoon) or a start time and an end time (e.g. 17:30-18:00). If the activity must occur at a specific schedule, the start time is only indicated. Multiple intervals can be specified in a single day provided they do not overlap in pairs. We give in Table A-1 some examples corresponding to the specification of two intervals in a day.

It is possible to extend this language to allow durations of several days and thus intervals which are defined on several days. Repetitions are then expressed over weeks or months. In Table A-2, an activity is specified for the maintenance of public spaces with a duration of two days each month except for august. The same activity is then specified but now only for the months 3, 6, 9 of years 2015 and 2016. In both cases, it is necessary to be able to point out a month number in the year or a day number in the month with the functions monthyear() and daymonth().

Table A-1: Examples of interval specifications

Specification	Semantic
2 times	2 interventions in the day at any moment
morning afternoon	1 intervention in the morning and 1 in the afternoon
morning 15:00-17:00	1 intervention in the morning and 1 between 15:00 and 17:00
8:00-10:00 15:00-17:00	1 intervention between 8:00 and 10:00 and 1 between 15:00 and 17:00
9:00 16:00	1 intervention at 9:00 and 1 at 16:00

```
Expressions
<simple-temporal-expression> ::= <period> <days> <intervals>
<general-temporal-expression> ::= {<simple-temporal-expression>}+

Period
<period> ::= <starting-date> "-"{<ending-date>}?
<starting-date> ::=/a date with the format mm/dd/yy/
<ending-date> ::=/a date with the format mm/dd/yy/

Days
<days-expression-form1> ::= {<date>}+ | "holiday" | {<date>}+ "holiday"
<days-expression-form2> ::= {<day-of-week>}+ ["except("{<date>}*
      ["holiday"]")"]
<days-expression-form3> ::= {"everyday" | "every(n)days("<date>")"}
["except("{<date>}* {<day-of-week>}* ["holiday"]")"]
<date>::=/a date with the format mm/dd/yy/
<day-of-week>::= "monday"|"tuesday"|"wednesday"|"thursday"|"friday"|
      "saturday"|"sunday"

Intervals
<intervals>::= (<integer> "times") |{<interval>}+
<integer>::= /number of occurrences of the activity in the day/
<interval>::=<string-form>|<pair-form>|<hour-form>
<string-form>::="morning"|"midday"|"afternoon"|"evening"|"night"
<pair-form>::=<starting-hour>"-"<ending-hour>
<hour-form>::=<hour> /an hour with the format hh:mm
```

Listing A-1: BNF syntax for our temporal language

Table A-2: Specification for an activity that lasts several days

Activity	Period	Months	Intervals
Maintenance of public spaces	01/01/15-12/31/16	Every month except(august)	daymonth(1)-daymonth(2) except(holiday)
		august	daymonth(1) except(holiday)
Maintenance of public spaces	01/01/15-12/31/16	monthyear(3) monthyear(6) monthyear(9)	daymonth(1)-daymonth(2) except(holiday)

AUTHOR BIOGRAPHIES

Dr. Marinette Bouet is an associate professor at the University of Clermont-Ferrand. She conducts her research in the world of databases. Specifically, she is interested in content-based image retrieval in image databases and in image mining. Most recently, she has become interested in data and service integration and home care management.

Dr. Michel Schneider is emeritus Professor of Databases and Information Systems at the University of Clermont-Ferrand and has over 35 years of experience as lecturer and researcher in the information technology field. He has contributed to many different research topics including: decision support systems, data warehouses, mediation systems and semantic models for databases. He continues to participate in several editorial boards covering these topics.