Implementation of a Distributed Guideline-Based Decision Support Model within a Patient-Guidance Framework

Erez Shalom[1], Yuval Shahar[1], Ayelet Goldstein[1], Elior Ariel[1], Moshe Sheinberger[1], Nick Fung[2], Valerie Jones[2], and Boris Van Schooten[2]

[1]The Medical Informatics Research Center, Department of Information System Engineering, Ben Gurion University of the Negev, Israel
{erezsh, yshahar,gayelet,eliorar, sheinmos}@bgu.ac.il
[2]University of Twente, The Netherlands
{l.s.n.fung, V.M.Jones}@utwente.nl, b.vanschooten@rrd.nl

Abstract. We report on new *projection engine* which was developed in order to implement a distributed guideline-based decision support system (DSS) within the European project MobiGuide.In this model, small portions of the guideline knowledge are projected, i.e. 'downloaded', from a central DSS server to a local DSS in the patient's mobile device, which then applies that knowledge using the mobile device's local resources. Furthermore, the projection engine generates guideline projections which are adapted to the patient's previously defined preferences and, implicitly, to the patient's current context, which is embodied in the projected knowledge. We evaluated this distributed guideline application model for two complex guidelines: one for Gestational Diabetes Mellitus, and one for Atrial Fibrillation. We found that the initial specification of what we refer to as the *customized* guideline should be in the terms of the distributed DSS, i.e., include two levels: one for the central DSS, and one for the local DSS. In addition, we found significant differences between the customized, distributed versions of the two guidelines, indicating further research directions and possibly additional ways to analyze and characterize guidelines.

Keywords: clinical guidelines, decision support, distributed computing.

1 Introduction

1.1 The need for distributed decision support

Clinical Guidelines (GLs) are a well-established method for enhancing the quality of care and for reducing costs [1]. Usually, the GL's recommendations are addressed solely to the care providers, such as the physicians, and not to the patients, and typically at the point of care, and not at home. Thus, existing frameworks for providing automated GL-based decision support focus mainly on supporting the care providers at the point of care. However, the role of the patient in the process of care is becoming more and more central. In addition, many GL recommendations address patient behavior, especially in the case of chronic illnesses, where treatment must be continued outside the hospital and partly managed by the patient. Therefore we believe that patients, and in particular, chronic patients, should be empowered to manage their own disease by extending both the GLs and the GL-based decision-support frameworks to provide guidance for patients outside the standard clinical settings. An architecture for patient guidance could provide the patients themselves with appropriate GL-based alerts and recommendations, and could also monitor and react to changes in the patient's personal environment. Both objectives can be achieved through the use of applications running on mobile devices. The potential of mobile devices for assisting patients in the process of self-care has already been demonstrated; one compelling example relates to the goal of improving adherence to taking medications at home [2]. Ideally, recommendations should be personalized, in the sense of considering the patient's personal schedule, important external events, and personal preferences corresponding to changing contexts. Such personalization can be achieved through an extension of the GL, customizing it to consider explicitly non-clinical contexts that were not accounted for in the original GL, such as the patient living alone, or the battery status of the mobile device [3].

However, monitoring alone is insufficient; mobile-based applications also need to factor in patient education as well as up to date, real-time GL-based recommendations [4, 5]. In addition, the full GL, which might need to be frequently updated, and the patient's full medical history, might need to reside on a central server, which will have a complete view of the patient's course of disease, as well as of the relevant clinical knowledge. Connection to such a server cannot always be guaranteed; the server may become unavailable due to an unexpected overload or other technical factors. In addition, some GL-based tasks should be delegated to the mobile device as a matter of course to prevent overburdening the central server. This can be achieved by distributing commonly occurring, computationally intensive tasks, especially when based on high-frequency data, such as continuous monitoring and detection of cardiac arrhythmias in patients prone to such a disorder, close to the patient. An example of such an intense calculation is the detection of potential patterns of Atrial Fibrillation (AF) episodes in an individual patient, based on a set of high frequency ECG sensor signals. Thus, *there is a need to distribute the decision support process from the central server to a local mobile device*.

1.2 The distributed decision support model in the MobiGuide project

To address the challenges associated with real-time patient guidance systems, the European Union's MobiGuide project [3] was initiated. The main goal of this project is to develop a distributed patient guidance system which integrates historical hospital records and current monitoring data into a Personal Health Record (PHR) accessible by patients and physicians, and providing personalized, secure, clinical-guideline-based guidance both inside and outside standard clinical environments. The distributed model of such a framework might be implemented as a service oriented architecture [6], which might be more suitable for distributing a process inside a hospital. However, in the case of the MobiGuide project, we have chosen to split the architecture into two main components: a *back-end Decision Support System* (*BE-DSS*) residing on a server system (this could be a cloud server, or, as in our case, on-premise servers in hospitals), and a *mobile DSS* (*mDSS*) residing on the patient's mobile device. The local mDSS is necessary to distribute computationally intensive monitoring and decision-making processes, with respect to data and knowledge requirements, at the local device level.

Previously [7], we had presented very briefly a new model we had developed for a distributed DSS, which we have implemented within the MobiGuide framework: In this model, portions of the clinical *guideline* (*GL*) which can be identified as a self-contained executable knowledge packages to be potentially applied in the mDSS, are projected to the local mDSS. To the best of our knowledge, only the GLARE GL application framework [8] introduced explicitly the concept of distributed GL-based decision support, implemented by managing several agents that interact in different clinical settings, called "contexts". However, that extension was intended to deal mostly with human interaction and communication, and with human resources management; the agents were human; and none of the agents mentioned was the *patient*.

Figure 1 demonstrates the MobiGuide projection workflow model: After the physician initiates the application of the *Gestational Diabetes Mellitus* (*GDM*) guideline (number 1), the BE-DSS retrieves the full GDM GL from the GL *knowledge-base* (KB) server (number 2), but sends only the specific sub-plan "monitor blood glucose once a week", which was previously tagged as a *projected plan* , to the mDSS; this sub-plan (representing the current treatment plan for that individual, and personalized with patient preferences and current context) is then applied by the mobile device [9] (number 3). At any time, a certain predefined *breakout* temporal pattern might be detected by the mDSS (as part of the projected sub-plan) or by the BE-DSS. When a breakout pattern is detected by the mDSS, the mobile device sends a message to the BE-DSS to "take control" and continue the application of the GL, indicating which pattern caused the breakout. This message to the BE-DSS from the mDSS is called a *callback*, and is also predefined within the projection (number 4). A callback, or detection of such a pattern by the BE-DSS, will in some circumstances lead to the sending of a new projection file by the BE-DSS to the mDSS (number 5) (e.g. effecting a change in the treatment plan).

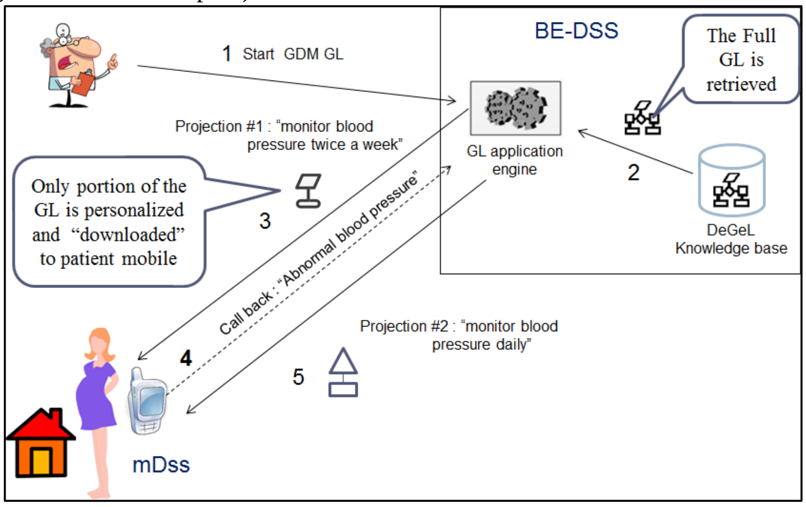

1 Start GDM GL
BE-DSS
The Full GL is retrieved
Projection #1 : "monitor blood pressure twice a week"
GL application engine
2
Only portion of the GL is personalized and "downloaded" to patient mobile
3
DeGeL Knowledge base
Call back : "Abnormal blood pressure"
Projection #2 : "monitor blood pressure daily"
4
5
mDss

Fig 1. An example of a projection model workflow.

For example, the breakout pattern “abnormal blood-pressure” detected by the mDSS (or BE-DSS) might lead to stopping the “twice a week blood-pressure measurement” sub-plan, and starting a new “daily blood-pressure measurement” sub-plan. In both cases, the switch between the current sub-plan and the new sub-plan, occurs at the BE-DSS level, which sends a new projection file to the mDSS. A projection file includes one or more projected plans, which will be activated immediately and applied by the mobile device. Note that The BE-DSS can manage, when necessary, the patient in centralised fashion, but it does not use the projection and callback mechanisms for that purpose. The BE-DSS sends the mobile device’s API a direct notification with the recommendation details, including a potential interaction with the patient. The notification bypasses the projection and callback mechanisms.

1.3 Aspects affecting the distribution of decision-support

The decision which actions or overall management plans should be distributed to a set of mobile devices, and which need to stay centralized, is not a trivial one, and is affected by several factors, as we have previously pointed out [7].
For example, consider the clinical task of detecting cardiac arrhythmias, such as AF.. We can continuously monitor each patient’s high-frequency ECG sensor signals by means of patient-worn biosensors linked via Bluetooth to the mobile device, and detect a pattern of AF that can be abstracted from these signals by the mDSS and sent to the central PHR to support a guideline-based recommendation to the patient and/or physician by the BE-DSS. Such a distribution of labor, in which the AF detection for each patient is performed locally by the mDSS is natural, and prevents an over burdening of the central BE-DSS server. Furthermore, the local mDSS is also essential for continuity of care when for some reason there is no internet connection to the central DSS; we would still want the patient to be provided with alerts relevant to the latest guideline by which she is being managed.

However, not all decisions can be made locally on the mobile (despite the ever increasing processing power and storage capacity of smartphones). Some decisions require the full historical (longitudinal) patient medical record, including all laboratory tests, physical examination, diagnoses, hospitalizations, procedures, and other interventions, which for practical reasons cannot, and for security and privacy reasons should not, reside on the mobile device. Furthermore, in some cases, the decision to be made is part of a long-term plan in the complete clinical guideline knowledge base (which is continuously maintained and updated as needed by medical domain experts), and in other cases, broader medical declarative knowledge (and interpretation) may be needed in order to detect a meaningful clinical pattern which may require a switch to another branch in the guideline, or even to a completely different guideline; such knowledge resides only on the central knowledge-base server. In such cases, as described above, we wish the mDSS, when encountering certain local predefined (temporal) patterns, to send a callback message to the BE-DSS, asking for further instructions, resulting in a BE-DSS recommendation, and possibly even in a completely new projected guideline or guideline branch.

For these reasons we adopted a distribution policy based on a number of factors. The policy is applied starting at the stage of medical knowledge engineering (i.e., when customizing the GL for distributed care), in order to determine whether various decisions and actions should be applied at the BE-DSS level or at the mDSS level, following the general principles that we proposed in [7]. The factors are: the actor of the decision (patient or physician); the temporal horizon of future recommendations (e.g. immediate alerts by the mobile device when some value is out of range, versus longer-term guideline-based decisions made at the server); the data and knowledge resources needed for the decision; the need for PHR access, and a consideration of where a potential personalization of the guideline should reside. These principles need to be considered by the knowledge engineer and the expert physicians during the knowledge specification phase. However, once it is decided that a decision can be applied by the mDSS, i.e., can support the patient when he/she is not with the physician, the relevant guideline knowledge needs to be projected to the mDSS.

In the sequel we describe how the projection model was implemented in the context of MobiGuide project, and how it was evaluated in the case of two complex GLs for management of GDM and AF. In general, the implementation of the projection model includes two main tasks: specification of the GL in terms of a distributed DSS, and development of the projection engine as part of the GL application engine; one of the subtasks of that engine is generating and personalizing the projections. In the following sections we will describe how we implemented these two tasks.

2 Specifying the Guideline in the Terms of a Distributed DSS

2.1 Choosing projected plans in the GL

Specifying a GL for application by a fully distributed DSS requires a different strategy from traditional GL knowledge engineering. As explained above, it involves multiple new challenges, such as: deciding at which level (mDSS or BE-DSS) each plan or decision should be placed, deciding which action or plan needs to be performed, and deciding which breakout patterns should trigger callbacks to the BE-DSS. This process is performed during the GL specification phase by a knowledge engineer in collaboration with expert physicians, as part of the process of creating a consensus regarding the GL [10].

Based on our experience, we have outlined several principles for selecting which plans in the GL should be projected [7], some of which were mentioned in Section 1.3. These characteristics helped us understand when a certain decision task should be delegated to the local mDSS, and when it should best be left to the central BE-DSS. In general, computations that are intense, though not necessarily algorithmically challenging, and that can be easily distributed and performed using only local patient data, such as AF detection given only a single patient's data, should be distributed to the multiple local devices. Computations that are algorithmically complex (e.g., require complex temporal pattern detection) and that require the patient's historical, longitudinal medical record, and/or the full knowledge base, should be performed centrally.

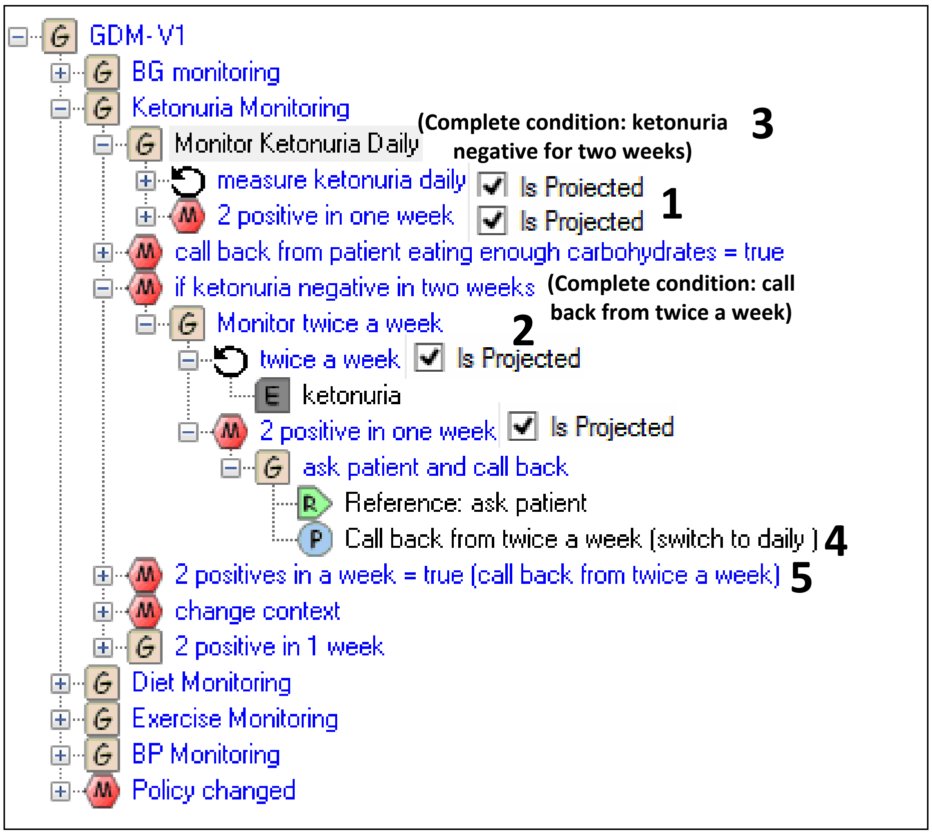
Table 1 shows several examples of the distribution of decision making between the BE-DSS and the mDSS for the GDM and AF GLs.

Table 1. Examples for choosing decisions at the different DSS levels

Domain	Level of decision	Decision description	Explanation
AF	BE-DSS	Is the patient eligible for the "Pill-in–the-pocket" emergency plan?	The relevant data are stored in the PHR and are therefore not accessible by the mobile device
GDM	mDSS	"when a pattern of two weeks of normal blood-glucose is detected, send a callback to BE-DSS "	Requires only a simple calculation based on relatively short-term accumulating daily blood-glucose values
AF	mDSS	Monitor AF episodes	Abstracted from high-frequency ECG signals generated by a local sensor whose data are accessible to the mobile; requires intensive computation using local data, which is best performed locally for each individual patient
GDM	BE-DSS	Does the Ketonuria abstraction have a negative value over the past week AND the Diet has not been changed since the last visit?	The data about visits to clinicians resides in the PHR, accessible to the back-end DSS, and do not exist locally in the mobile device
GDM and AF	BE-DSS	Context change	Context change is affecting multiple GL plans and requires a global view of the complete guideline

Figure 2 shows how we implemented the tagging of plans as plans to be projected as part of the GL, using the GESHER knowledge acquisition tool [11], as part of the *Digital electronic Guideline Library* (DeGeL) [12], in the case of the GDM GL: At specification time, the knowledge engineer checked the "*is-projected*" property of the sub-plans that were determined as sub-plans that need to run at the mDSS level, in this case the "monitor Ketonuria daily" sub-plan (number 1), and the "monitor [for Ketonuria] twice a week" sub-plan (number 2). Note that in both cases, two sub-plans are tagged as *projected*: the first is a periodic sub-plan for measuring the ketonuria each day (the circular arrow shape); the second is a monitoring sub-plan (the hexagonal shape), which in fact monitors for a breakout pattern (in this case, the pattern "two positive values in a week of ketonuria"), and which, if detected, asks the patient a question regarding their diet and triggers a callback to the BE-DSS to determine how to proceed. Note that several projected plans (depending on their internal eligibility criteria and the GL's overall workflow) might be sent in the same projection file to the mobile device. The basic language used for representing the GL, underlying the

GESHER GL-specification tool, is Asbru [13], and its hybrid-Asbru extensions [12]; we have augmented it using the projection and callback tags.



```
GDM- V1
    BG monitoring
    Ketonuria Monitoring
        Monitor Ketonuria Daily   (Complete condition: ketonuria   3
                                    negative for two weeks)
            measure ketonuria daily   ☑ Is Projected
            2 positive in one week    ☑ Is Projected   1
        call back from patient eating enough carbohydrates = true
        if ketonuria negative in two weeks  (Complete condition: call
                                             back from twice a week)
            Monitor twice a week                2
                twice a week  ☑ Is Projected
                    ketonuria
                2 positive in one week  ☑ Is Projected
                    ask patient and call back
                        Reference: ask patient
                        Call back from twice a week (switch to daily )  4
        2 positives in a week = true (call back from twice a week)  5
        change context
        2 positive in 1 week
    Diet Monitoring
    Exercise Monitoring
    BP Monitoring
    Policy changed
```

Fig 2. The GESHER interface and the tagging of projected plans in the GL, in the case of the GDM GL

2.2 Definition of appropriate callbacks

As explained above, projections and callbacks support a continuous dialog between the BE-DSS and the mDSS. Thus, the GL created to support a distributed DSS is also specified in terms of messages between the mDSS and the BE-DSS. This includes specifying the projections to send the mobile device, the breakout patterns to be detected by the mDSS, and the associated callbacks from the mDSS to the BE-DSS. On the other hand, at the BE-DSS level, monitoring plans are "listening" to all of the relevant breakout patterns, and to callback messages coming from the mDSS, which might cause the BE-DSS to send the mDSS a message to stop an existing projected plan, or to send a new projected plan to be activated.

The implementation of this unique dialog in terms of GL specification is shown also in Figure 2: first, "monitor ketonuria daily" is projected to the mobile (number 1), and accordingly a monitoring sub-plan to detect the temporal pattern "ketonuria has been negative for two weeks" is activated, in this case, at the BE-DSS level, as determined by the knowledge engineer who built the parallel workflow for this section of the GL (note that in this particular case, it could also be applied by the mDSS by projecting it to the mobile device). When that sub-plan is triggered (by another part of the BE-DSS, an *intelligent monitoring* module that monitors the data for knowledge-based temporal patterns and that is subscribed to the urine measurements reaching the PHR server (see section 3, number 2), two events occur: 1) The *complete condition* (in the terms of the Asbru language [13], in which MobiGuide GLs are specified) for the daily monitoring sub-plan is triggered, thus causing the BE-DSS to send a projection to the mDSS to stop it (number 3), and 2) a new sub-plan, to reduce the frequency of monitoring to twice a week is started and is projected by the BE-DSS to the mDSS.

Note that the new projected "monitor ketonuria twice a week" sub-plan, which replaces the originally projected "measure ketonuria daily" sub-plan, includes a call-back instruction (number 4) to the BE-DSS, in case the mDSS detects the breakout pattern of two positive values of ketonuria in a week. When the mDSS detects this breakout pattern, a callback is sent to the BE-DSS. This call-back is constantly monitored (through the intelligent monitoring module) by a specific monitoring sub-plan (number 5); thus, when the callback arrives at the BE-DSS, it causes the BE-DSS to send a stop message regarding the sub-plan for twice weekly monitoring (number 2), and to project to the mDSS a daily monitoring plan.

3 The projection engine

In order to support the distributed DSS projection model, we developed a new component, the *projection engine*, which extends the functionality of our existing GL application engine [14]. The extended BE-DSS architecture is shown in figure 3: the GL application engine gets the GL knowledge from the GL KB, and applies it. When the GL application engine finishes, the projection engine examines which parts of the existing activated sub-plans need to be projected. The projection engine then retrieves the preferences and personalized contexts [3] of the patients from the data integrator through the data and knowledge services layer, and may also perform queries via the intelligent monitoring module (e.g., to get the current context of the patient). Then, the projection engine generates the projection file, which is sent by the GL application engine to the mDSS through the *Body Area Network* (BAN) back-end server [15], which mediates between the BE-DSS and the mDSS.

The projection engine produce two types of projections: (1) Declarative projections including concepts (simple abstractions), and personal (patient-specific) events that induce predefined customized contexts in the GL, within which the guideline's actions might be modified, and (2) Procedural projections – including sub-plans for general treatments or for specific treatments relating to personalized-contexts.

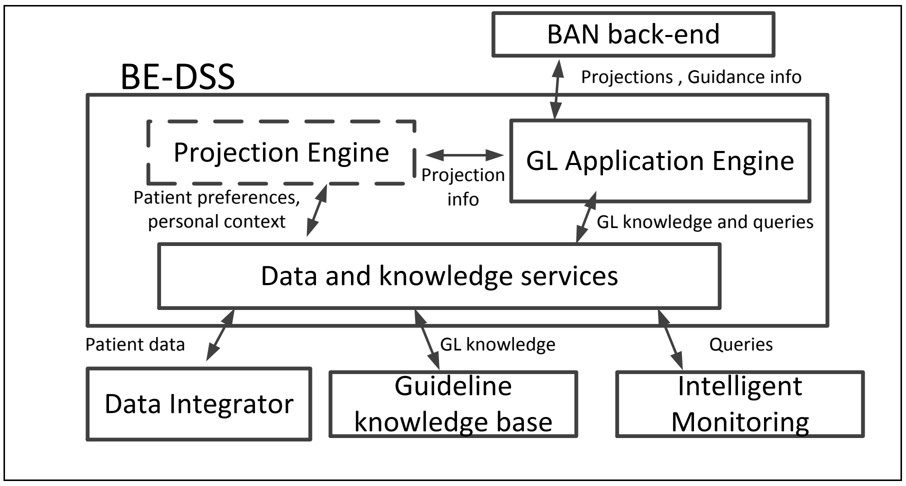


Fig 3: High level architecture of the BE-DSS and the projection engine

3.1 Declarative projections

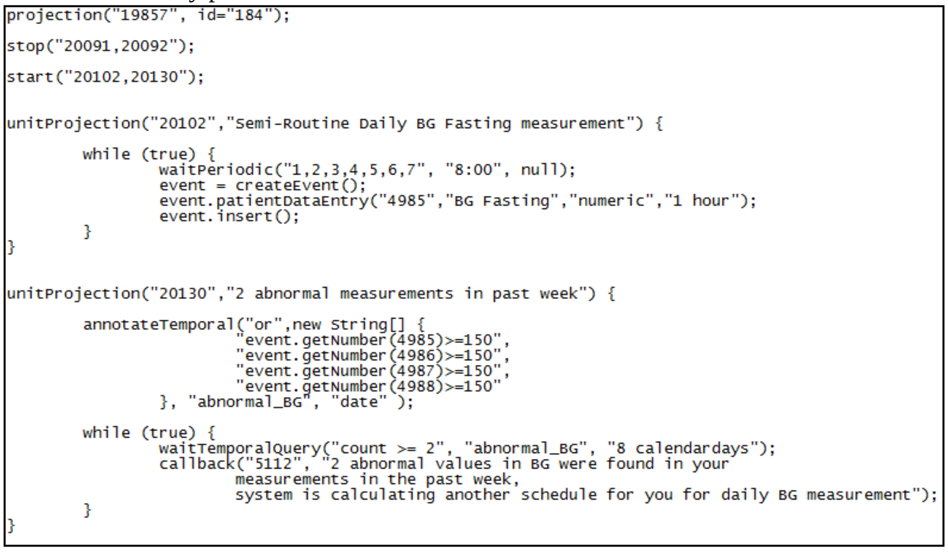
One of the innovative features of the MobiGuide system is support for personalization. Personalization of the GL occurs when the patient is enrolled as a user of the MobiGuide system. In this step, predefined clinical contexts, already part of the customized GL (e.g., a High Carbohydrate Meal context, or an Irregular Schedule context), are retrieved from the knowledge base, and are shown to the patient so that he or she can choose the corresponding personal events that induce these predefined contexts (e.g. "Wedding” or "Vacation"). The mapping between the personal events and the contexts is stored in the PHR and is sent to the mDSS before starting the GL application session, as part of the declarative projection. The mDSS uses the declarative projection to send to the patient’s interface the list of personal events that were selected during the initial enrollment session as inducing certain predefined contexts in the customized guideline. For example, for a given patient, "Vacation" or “Holiday” events might induce the predefined context “irregular schedule”, and a wedding event might induce the predefined context “High Carbohydrate Meal”. For example, a patient might register on her smartphone the personal event "at work", the mDSS reports this event to the PHR. The event, for her, induces (at the level of the BE-DSS) the “Regular-Schedule” context. If this patient subsequently reports registers the personal event "holiday," this will be then reported by the mDSS to the PHR, inducing, for her, (at the level of the BE-DSS) the context “Irregular Schedule”, thereby leading the BE-DSS to send the mobile device a second projection to be applied by the mDSS, with a different periodic monitoring rate. The BE-DSS always generates the projections to be sent to the mobile device according to the current (possibly induced by a personal event) context of the patient.

The declarative projection also includes the *Quality of Data* (QoD) [16] information used by the *Quality of Data Broker* located at the mobile, in order, for example, to “ignore” invalid input data (e.g. an out of range blood pressure value).

3.2 Procedural projections

Procedural projections are generated at run-time by the projection engine. The engine checks the "*is projected*" property for each sub-plan (see figure 2) , and if it is set to "true", the sub-plan's corresponding projection file is generated and added to the projection collection (see below for more details about projection collection). Otherwise, the BE-DSS engine continues to apply the sub-plan. Also, as part of the projection process, the specific thresholds of the patients (e.g. personal target exercises levels), and the preferences relating to the personal contexts of the patient are retrieved from the PHR and are set in the projections.

Each projection file is decomposed into several "unit-projections". Each unit-projection is a single sub-plan. For example, the sub-plan "monitoring blood glucose once a week" is decomposed into two unit-projections: 1) the sub-plan for blood glucose measurement schedule, and 2) the sub-plan to monitor several days of high fasting blood glucose levels, signifying that the patient is not well controlled. Each projection and each unit within a projection starts independently, as soon as it arrives at the mobile device. Each unit has its own set of internal temporal constraints, including temporal relations among different actions. Thus, all important temporal-constraint knowledge resides (is encapsulated) within the projected units and not among them, hence concurrency problems are not an issue.



```
projection("19857", id="184");

stop("20091,20092");

start("20102,20130");


unitProjection("20102","Semi-Routine Daily BG Fasting measurement") {

        while (true) {
                waitPeriodic("1,2,3,4,5,6,7", "8:00", null);
                event = createEvent();
                event.patientDataEntry("4985","BG Fasting","numeric","1 hour");
                event.insert();
        }
}


unitProjection("20130","2 abnormal measurements in past week") {

        annotateTemporal("or",new String[] {
                        "event.getNumber(4985)>=150",
                        "event.getNumber(4986)>=150",
                        "event.getNumber(4987)>=150",
                        "event.getNumber(4988)>=150"
                }, "abnormal_BG", "date" );

        while (true) {
                waitTemporalQuery("count >= 2", "abnormal_BG", "8 calendardays");
                callback("5112", "2 abnormal values in BG were found in your
                        measurements in the past week,
                        system is calculating another schedule for you for daily BG measurement");
        }
}
```

Fig 4. An example of a projection file sent to the mDSS from the BE-DSS, containing two unit-projections

Before building a projection on the fly, the projection engine checks what is the current context of the patient, (e.g. "Regular Schedule", which is induced by the "at work" event); it then retrieves all of the patient's scheduling preferences for this context (e.g., days and hours of reminders), and modifies the projections accordingly. For example, in unit-projection "20102", the time to activate reminders to the patient is set to "8:00" which is the preferred hour by the patient to get reminders in the context "Semi-routine Schedule".

In addition, the projection file contains two lists of IDs: one for the sub-plans to be stopped, and one for the sub-plans to be started. When the BE-DSS is triggered (e.g. by an incoming callback from mDSS, or through detection of a breakout pattern), all affected sub-plans which are needed to transit into their complete state are aggregated by the projection engine into a unified *stop-list*. On the other hand, sub-plans that need to start are aggregated into a *start-list*. Thus, the mDSS is always "up-to-date" with respect to the sub-plans to be stopped, or to be started at the local level (the rest of the plans currently applied by the mDSS are assumed by default to be continuing).

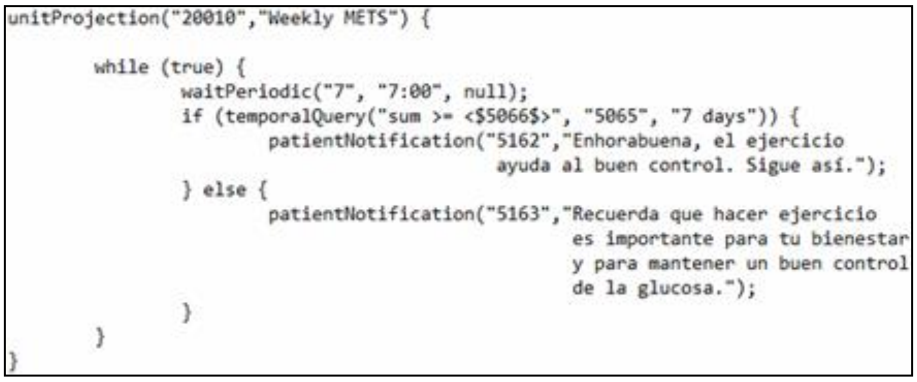
If the patient's mobile device crashes, the projection engine recovers the last procedural projections sent to the mobile device, and resends them to the mDSS. To support this functionality, we added to the BE-DSS a new *projected-plans* collection to store the different projections generated during the GL application session by the projection engine. Table 2 shows this collection in the context of the projection shown in Figure 2. Each row represents a unit-projection in the collection which has a link to the projection ID it belongs to, a unit-projection ID, the timestamp showing when it was sent to the mDSS, and status (started or stopped). Note that all unit-projections shown in Table 2 are sent from the same projection at the same time. The mDSS uses these properties to manage the execution of the sub-plans running locally (for example, it might stop the "daily blood pressure monitoring" sub-plan #22 from the stop-list, and start a new plan for "twice a week" monitoring sub-plan #23 from the start-list).

Table 2. The *projected-plans* data structure, which stores the different unit-projections generated by the projection engine

Projection ID	Unit-projection ID	SentDate	Status
184	20091	10/5/14/14:00:00:00	stop
184	20092	10/5/14/14:00:00:00	stop
184	20102	10/5/14/14:00:00:00	start
184	20130	10/5/14/14:00:00:00	start

3.3 Implementation of personalization with dynamic projections

At projection time, except for patient preferences for days and hours of reminders, the projection engine replaces all pre-defined knowledge thresholds with real values. An example of a knowledge threshold is the personal target level for physical exercise. Values above this threshold are abnormal. As the threshold values might be changed from time to time, writing their explicit value in the projection file will be hard to maintain. Instead, the threshold knowledge ID is written as a variable name (a string) circumscribed by triangular brackets; at projection time, the variable is replaced by the real value from the knowledge base. Figure 5 shows an example for unit-projection "20010": at design-time, the threshold knowledge ID is set, for example, to the variable "<\$5066\$>", (which in this case denotes the exercise target level). At projection time, the projection engine replaces this string with the real value; in this case, all thresholds are set to "5".



```
unitProjection("20010","Weekly METS") {

        while (true) {
                waitPeriodic("7", "7:00", null);
                if (temporalQuery("sum >= <$5066$>", "5065", "7 days")) {
                        patientNotification("5162","Enhorabuena, el ejercicio
                                        ayuda al buen control. Sigue así.");
                } else {
                        patientNotification("5163","Recuerda que hacer ejercicio
                                                es importante para tu bienestar
                                                y para mantener un buen control
                                                de la glucosa.");
                }
        }
}
```

Fig 5. The knowledge thresholds in the case of calculating the threshhold for weekly exercise. Patient notification texts are shown in Spanish (for the GDM pilot in Spain)

Another dynamic projection behaviour is handling drug prescriptions: as drug prescriptions are patient-specific they cannot be part of the GL knowledge. Thus, the projection engine adds each valid drug medication (plus its appropriate dose and schedule) it finds in the patient's personal record dynamically as a unit-projection, and converts the dates to start and stop the drug to total days. Each drug is then personalized according to the current context of the patient, for example, in the case that the time for taking a drug is set to "after lunch", this time is generated according to the lunch hour belongs to the current context so that the mDSS receives the specific hour for taking drug. In addition, the projection engine add a drug to the stop-list it is not valid anymore..

3.4 Evaluation of the projection model

Together with expert-physicians we specified the GDM [17] and AF GLs using the GESHER knowledge acquisition tool [11]. A complete discussion of the knowledge acquisition process is out of the scope of this paper. Each GL took approximately 3 months to specify in detail, in collaboration with domain experts.
Following that phase, we identified projected plans in the GL. Most of the projected plans were periodic and monitoring sub-plans. Projected periodic sub-plans are plans in which some action should be performed periodically by the mDSS (see section 2). Table 3 shows the distribution of the projected plans between the GLs, and their characteristics. Altogether, we tagged 39 projected plans to the GDM GL, and 20 for the AF GL. Note that in the case of the AF GL, most of the projections were of periodic sub-plans and not of monitoring sub-plans, and there are only 2 callbacks. That is because most of this GL's actions can be handled by the mobile device, which thus very rarely needs the BE-DSS to change the projection. In contrast, in the case of the GDM GL, more decisions are made at the BE-DSS level, since more decisions in that GL require additional data (such as past and future visits) and care-giver confirmations, both of which are accessibly only to the BE-DSS.

Table 3. **The distribution of the projections** projected plans **between the GLs**

	projected Periodic sub-plans	projected Monitoring sub-plans	Callbacks	Decisions made at BE-DSS	Decisions made at mDSS
GDM	22	17	16	44	34
AF	18	2	2	24	36

4. Summary and Discussion

We have presented an innovative framework for the distributed application of clinical guidelines through a two-tiered architecture, which integrates a central DSS server with multiple local mobile devices that monitor individual patients, and thus splits the computational tasks of applying a GL between them. The projection and callback mechanism that we have implemented support a continuous dialog between the BE-DSS and the mDSS, splitting the decision-support tasks between the central BE-DSS, which is linked to the overall medical knowledge base and to the patient's EMR, and the local mDSS, and exploit the relative advantages of the different computational architectures and their respective access to clinical data and medical knowledge.
To the best of our knowledge, the distributed GL application architecture that we have designed and implemented is entirely new; even previous studies suggesting the distribution of GL application, mostly referred to the assignment of different tasks to different [human] agents, per their specialty [8]. We believe that the principles underlying this two-tiered architecture are rather general, and support both functional (e.g., send recommendations to patient) and non-functional (e.g., efficiency, security) requirements.

We have implemented both the distributed GL specification and the distributed GL application in the case of the GDM and AF guidelines. We found that these two GLs create very different projection and callback profiles when represented as distributed GLs. The difference might represent a profound difference between the characteristics of the two GLs, possibly due to the increased need of accessing the EMR for the patient's history, in the case of the GDM GL. Representing additional GLs in a distributed format might lead to a better understanding of GL characteristics and to additional insights regarding the differences amongst them.
We are in the final year of the four year MobiGuide project. and are currently running a pilot study to test the feasibility of using a distributed DSS architecture to manage patients using the two guidelines: The GDM guideline, applied in collaboration with the Sabadell Hospital in Barcelona, Spain, and the AF guideline, applied in collaboration with the Fondazione Salvatore Maugeri, Pavia, Italy.

Acknowledgements. The MobiGuide project (http://www.mobiguide-project.eu/) has received funding from the EU's Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 287811.

References

1. Quaglini S, Ciccarese P, Micieli G, et al. Guideline Application for Decision Making in Ischemic Stroke (GLADIS) Study Group. Stud. Health Technol. Inform. 101 (2004) 75-87

2. Granger BB, Bosworth HB. Medication adherence: emerging use of technology. CurrOpin-Cardiol. 2011 Jul; 26(4): 279-287
3. Peleg, M., Shahar, Y., Quaglini, S. Making healthcare more accessible, better, faster, and cheaper: the MobiGuide Project. European Journal of ePractice: Issue on Mobile eHealth 2014; 20, pp. 5-20
4. Chomutare T, Fernandez-Luque L, Arsand E, Hartvigsen G. Features of mobile diabetes applications: review of the literature and analysis of current applications compared against evidence-based guidelines. J Med Internet Res. 2011;13(3):e65. doi: 10.2196/jmir.1874
5. Farmer AJ, Gibson OJ, Dudley C, Bryden K, Hayton PM, Tarassenko L, Neil A. A randomized controlled trial of the effect of real-time telemedicine support on glycemic control in young adults with type 1 diabetes. Diabetes Care. 2005 Nov;28(11):2697-702.
6. Besana P, Barker A, Towards Decentralised Clinical Decision Support Systems.Advanced Computational Intelligence Paradigms in Healthcare 5,Studies in Computational Intelligence Volume 326, 2011, pp 27-44
7. Shalom E, Shahar Y, Goldstein A, et al. Enhancing Guideline-based decision support with distributed computation through local mobile application. Prohealth 2014
8. Bottrighi, G. Molino, S. Montani, P. Terenziani, M. Torchio, Supporting a Distributed Execution of Clinical Guidelines, Computer Methods and Programs in Biomedicine 112 (2013) 200-210).
9. Sacchi, L., Fux, A., Napolitano, C., Panzarasaa, S., Peleg, M., Quaglini, S., Shalom, E., Soffer, P., Tormene, P. (2013). Patient-tailored Workflow Patterns from Clinical Practice Guidelines Recommendations. Medinfo 2013
10. Shalom, E., Shahar, Y., Taieb-Maimon, M., Lunenfeld, E., Bar, G., Yarkoni, A., Young, O., Martins S.B., Vaszar, L.T., Goldstein, M.K., Liel, Y., Leibowitz, A., Marom, T., and Lunenfeld, E. (2008). A quantitative evaluation of a methodology for collaborative specification of clinical guidelines at multiple representation levels. *The Journal of BioMedical Informatics* 41(6), *889-903.*
11. Hatsek, A., Shahar, Y., Taieb-Maimon, M., Shalom, E., Klimov, D., Lunenfeld, E.: A Scalable Architecture for IncrementalSpecification and Maintenance of Procedural and Declarative Clinical Decision-Support Knowledge. The Open Med Info J. vol. 4 , 255-277 (2010)
12. Shahar, Y., Young, O., Shalom, E., Galperin, M., Mayaffit, A., Moskovitch, R., et al. A framework for a distributed, hybrid, multiple-ontology clinical-guideline library, and automated guideline-support tools. J Biomed Inform 2004;37(5):325-344.
13. Shahar, Y., Miksch, S., and Johnson P. The Asgaard project: A task-specific framework for the application and critiquing of time-oriented clinical guidelines. A I. Med. 14 (1998) 29-51.
14. Shalom, E., Shahar, Y., Parmet, Y., and Lunenfeld, E. A multiple-scenario assessment of the effect of a continuous-care, guideline-based decision support system on clinicians' compliance to clinical guidelines. The International Journal of Medical Informatics, in press. DOI: 10.1016/j.ijmedinf.2015.01.004.
15. Jones, V.; Bults, R.; Konstantas, D.; Vierhout, P., *Healthcare PANs: Personal Area Networks for trauma care and home care*, Fourth International Symposium on Wireless Personal Multimedia Communications (WPMC), Aalborg, Denmark, 2001.
16. Larburu, N., Van-Schooten, B., Shalom, E., Fung, N., Hermens, H., and Jones, V. A Quality Aware Mobile Decision Support System for Patients with Chronic Illnesses. Prohealth 2015.
17. García-Sáez, G., Rigla, M., Martínez-Sarriegui, I., Shalom, E., Peleg, M., Broens, T., Pons, B., Caballero-Ruíz, E., Gómez,E,J., and Hernando, M,E. "Patient-oriented Computerized Clinical Guidelines for Mobile Decision Support in Gestational Diabetes" Journal of Diabetes Science Technology, 2014 DOI: 10.1177/1932296814526492.