Using Semantic Components to Represent Dynamics of an Interdisciplinary Healthcare Team in a Multi-agent Decision Support System

Szymon Wilk[1], Mounira Kezadri-Hamiaz[2], Daniela Rosu[2], Craig Kuziemsky[2], Wojtek Michalowski[2], Daniel Amyot[2], Marc Carrier[3]

[1] *Poznan University of Technology, Poznan, Poland*
[2] *University of Ottawa, Ottawa, Canada*
[3] *Ottawa Hospital Research Institute, Ottawa, Canada*

Abstract

In healthcare organizations, clinical workflows are executed by interdisciplinary healthcare teams (IHTs) that operate in ways that are difficult to manage. Responding to a need to support such teams, we designed and developed the MET4 multi-agent system that allows IHTs to manage patients according to presentation-specific clinical workflows. In this paper, we describe a significant extension of the MET4 system that allows for supporting rich team dynamics (understood as team formation, management and task-practitioner allocation), including selection and maintenance of the most responsible physician and more complex rules of selecting practitioners for the workflow tasks. In order to develop this extension, we introduced three semantic components: (1) a revised ontology describing concepts and relations pertinent to IHTs, workflows, and managed patients, (2) a set of behavioral rules describing the team dynamics, and (3) an instance base that stores facts corresponding to instances of concepts from the ontology and to relations between these instances. The semantic components are represented in first-order logic and they can be automatically processed using theorem proving and model finding techniques. We employ these techniques to find models that correspond to specific decisions controlling the dynamics of IHT.

In the paper, we present the design of extended MET4 with a special focus on the new semantic components. We then describe its proof-of-concept implementation using the WADE multi-agent platform and the Z3 solver (theorem prover/model finder). We illustrate the main ideas discussed in the paper with a clinical scenario of an IHT managing a patient with chronic kidney disease.

Keywords: *interdisciplinary healthcare team; clinical workflow; multi-agent system; first-order logic; ontology.*

1. Introduction

Provision of automatic support for the execution of clinical workflows [1] is challenging in situations when these workflows include actions performed by an interdisciplinary healthcare team (IHT) [2] composed of practitioners with different specialties [3]. This challenge is twofold [4]: first, it is associated with dynamic role variability, which implies that team members may play a number of different professional roles that go beyond their specialties; and second, it is associated with clinical staff member variability, meaning that members may join and leave a team at different stages of workflow execution.

The multi-agent system (MAS) paradigm is well-suited to model and support workflow execution by an IHT because of the cooperative nature of the agents that "act" on behalf of the team members [5,6]. We used this paradigm to design and implement an earlier version of the MET4 system for supporting an IHT in workflow execution [7-9], where we focused on selected aspects of team dynamics (understood as team formation, management and task-practitioner allocation), mostly related to capability-based team member selection and task assignment. The initial implementation of MET4 included a scaled-down team ontology (described in [8]) and involved embedding algorithms for controlling a team's dynamics in agent plans [9]. While this initial approach was sufficient for supporting basic team operations it did not allow for more complex behaviors, especially related to the role of most responsible physician (MRP) being a team leader, and complex task assignment scenarios where dedicated team members should be selected for a number of different, but related tasks. We also wanted to improve scalability and adaptability of system architecture by creating a dedicated layer responsible for modeling team behavior.

While the MET4 system met our goal to support IHT management, it had to be extended in order to account for a wider spectrum of team dynamics. In this paper we describe how this extension was accomplished by: (1) expanding the ontology to represent more complex IHT characteristics; (2) translating the algorithms used in MET4 into external behavioral rules to enable better scalability and adaptability; and (3) verifying the proposed solutions in a comprehensive clinical scenario. Developing the extended MET4 system required introducing a new semantic layer based on first-order logic (FOL) formalism. This semantic layer allows for scalable modeling and controlling of team dynamics and consists of three *semantic components*: (1) a revised *ontology* describing concepts and relations pertinent to IHTs, executed workflows and managed patients, (2) *behavioral rules* governing the dynamics of an IHT, and (3) an *instance base* that stores facts representing the instances of concepts from the ontology and the relations between these instances. Thus, the instance base includes run-time contextual information about teams and their members, workflows, task assignments and patients. The semantic components may be automatically processed using theorem proving and model finding techniques [10]. More specifically, theorem proving is used for checking the applicability of the behavioral rules in a given context, and model finding is used to derive models corresponding to specific decisions related to the team's dynamics (e.g., assignment of a specific workflow task to a particular IHT member). These models include facts that may be stored in the instance base for future processing.

The extended MET4 was verified using a clinical scenario involving a patient with chronic kidney disease who is managed by an IHT according to a presentation-specific workflow. As part of this scenario, we provide a detailed description of the semantic components and their processing. We also describe operations of other components of the extended MET4 system, including specific agents, to show how they interplay with the semantic components and contribute to the improved IHT management support.

2. Related work

Our research addresses an important problem of modeling and supporting an IHT in execution of a clinical workflow. Thus, the related work can be divided into two research areas. The first area involves the work on the team-based modeling and execution of clinical workflows, while the second area is concerned with issues related to modeling the dynamics of an IHT, operationalizing it within the MAS paradigm as logic-based rules, and employing these rules together with reasoning techniques to control the execution of clinical workflows.

The first area of related research is well illustrated by formal and executable representations of clinical workflows and guidelines, involving languages and models proposed originally for business applications (e.g. BPMN [11,12]), as well as specialized languages developed for clinical problems (e.g. GLIF3, PROforma or SAGE – see their review in [13]). These specialized languages usually employ a task-network model [13] and assume a modeled workflow or guideline is a network of component tasks, such as data query, decision and action. Most of the representations were initially not well suited for team-based execution, and therefore several their extensions were proposed. For example, Müller and Rogge-Solti proposed "colored BPMN" where they tag tasks with colors corresponding to various professional roles of healthcare practitioners [14]. This allows for more concise team-based workflow models and for shared tasks that require multiple professional roles. However, the association of professional roles with tasks is static and needs to be explicitly stated when the workflow is being modeled. Grando et al. [15] proposed an extension of the PROforma language to handle two collaboration patterns – assignment and delegation of tasks. The major difference between these two patterns is the accountability for the task outcome – in the former case it is the practitioner that has been assigned a task, while in latter case it is the practitioner delegating a task. The authors also introduced the notion of skills, understood as a particular level of qualification or responsibility that are checked dynamically before task assignment or delegation. . While the approach by Grando et al. provides a insight into both collaboration patterns, it does not deal with the overall team management (e.g. selection and maintenance of a team leader).

The second area of related research can be further subdivided into: (1) representing the dynamics of an IHT, (2) using logic-based rules and reasoning in the MAS paradigm, and (3) employing MASs to support collaboration and execution of healthcare workflows. These lines of research are briefly discussed in subsequent paragraphs.

Considering the increasing complexity of patient conditions and the associated complexity of patient management, it is natural that the use of IHT in delivering care has attracted growing attention [16] and has resulted in ongoing research on IHT dynamics. One of the issues being explored is role blurring (see the review in [17]) that is defined as exchanging professional roles

by team members due to shared knowledge and skills which translates into role variability. Role blurring results in a number of positive outcomes like improved continuity of care or shared workloads. However, it may also result in issues, like team member anxiety if roles are poorly defined. Another topic associated with IHT dynamics is team leadership. According to the review in [18], a clearly defined leader positively affects team effectiveness and innovation. Moreover, due to professional hierarchy inherent in healthcare environment, there is a common expectation that teams are led by physicians – not responding to this expectation may result in frustration of other team members [18]. There is also existing research concerned with issues that are indirectly related to IHT dynamics, such as coordination of a team's activities and communication among the team members [19], the impact of team care delivery on patient safety [20], and improving the effectiveness of the teams' operations [21].

The use of logic-based rules in the MAS paradigm was discussed by Zambonelli et al. [22], who described how temporal logic can improve methodologies for the analysis and the design of MASs. These rules express general requirements for the execution of the system but do not allow for verification of these requirements. In [23], van der Vecht et al. proposed a modular reasoning model to allow for meta-reasoning about different organizational rules; however, a shortcoming of the proposed model is that it separates the organizational rules from the actual decision-making process. In [24] reasoning was used to control role enactment in an agent-based organization. Specifically, agents were able to reason about their capabilities (e.g., to achieve a goal or to execute an action). These capabilities were further checked by a designated gatekeeper before admitting agents to the organization where they played specific roles. Also, in the context of belief-desire-intention models [25,26] agents used rule-based reasoning for norm monitoring and norm-aware planning of their actions, where norms were understood as a standard of behavior shared by all agents.

In healthcare, MASs have been used for modeling group collaboration and awareness as part of implementing computer-supported cooperative work [27]. For example, the PalliSys system facilitated collection, analysis and access to palliative patient data [28], and MADIP allowed for the remote monitoring of patients' vital signs in an un-obstructive manner [29]. The MAS paradigm has also been used to develop systems supporting IHTs in executing clinical workflow and practice guidelines. Examples of these systems include HeCaSe2 [30], which supports practitioners through semi-automatic application of clinical practice guidelines for patients, K4CARE [6], which extended the functionality of HeCaSE2 by customization of general workflows for individual patients, and finally, the earlier version of the MET4 system [7-9], which provides basic support for managing IHT dynamics and workflow execution. The research described here extends the MET4 system to represent complex IHT dynamics as well as providing scalable strategies for controlling these dynamics.

3. Design and implementation of the extended MET4 system

We start this section with presenting the main requirements and assumptions related to the IHT dynamics and operations. They lay the foundations for the formal design of the extended MET4 system developed using the Organization-based Multi-agent System Engineering (O-MaSE) methodology [31]. O-MaSE starts with the specification of assumptions and requirements and iteratively translates them into a number of models that drive the system's implementation (the application of O-MaSE to designing clinical decision support system is presented in [32]). From the family of models associated with the O-MaSE methodology, we describe the two main models that define the system design – the *domain model*, which introduces the IHT ontology and behavioral rules, and the *agent class model*, which describes agent classes and system components such as repositories or external services.

3.1. Assumptions and requirements

The initial assumptions and requirements considered when developing the MET4 system are specified in detail in [9]. Here we focus on new and most relevant elements of the specification – Figure 1 provides an overview of important concepts and relations. We use different shades of gray in this figure to indicate the new concepts required by the extension described in this paper, the revised existing concepts, and the original existing concepts.

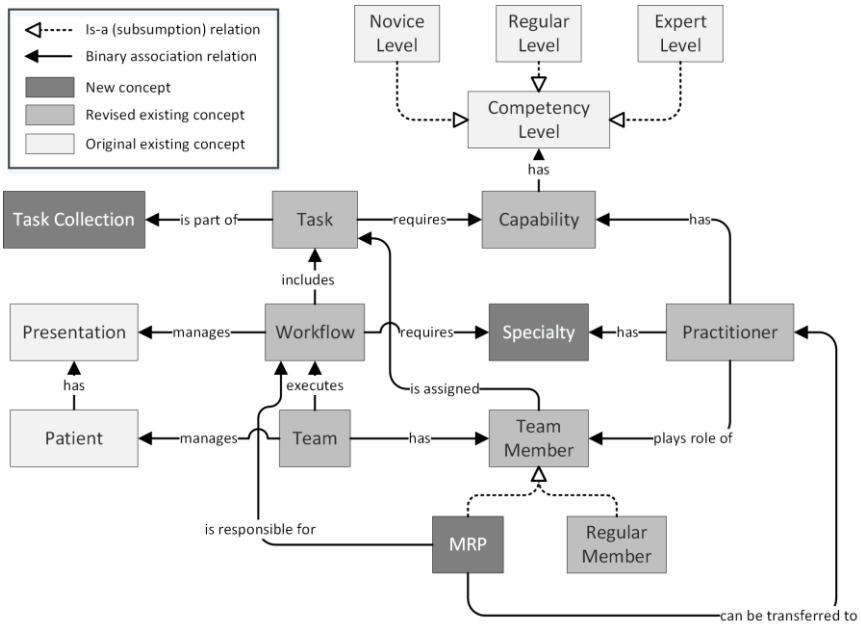

Figure 1 – Overview of the important concepts and relations

We assume a *team* is formed when a new *patient* with a certain *presentation* (e.g., a particular disease) is admitted and her management is initiated according to a presentation-specific *workflow*. *Practitioners* participating in a team may play two *team member* roles: a *regular member* and the *most responsible physician* (*MRP*). A regular member is a practitioner who executes *assigned tasks* from a workflow, while an MRP is a physician who is responsible for patient management, disposition decisions, dealing with exceptional situations, and patient discharge – and in that sense is a team leader. The concept of MRP replaces the concept of team leader previously used in [9] and responds to the general practice that IHT is led by a physician [18]. An MRP needs to be identified before the execution of the workflow starts and we assume that this role is assigned to the first available physician having the clinical *specialty* appropriate for the presentation-specific workflow. The role of MRP is relatively static as a physician stays with a team most of the time. However, this role can be temporarily *transferred* to another physician if a sub-workflow requiring another specialty is invoked. In contrast to a static nature of the MRP role, we assume that regular team members are recruited dynamically from among available practitioners and that they are released from the IHT after assigned tasks have been completed.

While information about practitioner's specialty is sufficient for MRP selection, it provides too coarse a characterization for flexible assignment of practitioners to workflow tasks. Such flexibility is required by the variability/blurring of professional roles associated with the *regular member* role in a team. Therefore, we also propose a fine-grained description of practitioners and tasks based on the *capabilities* for performing a task, with the associated *competency levels* being a proxy for skill levels (e.g., *novice level* vs. *regular level* vs. *expert level*). The recruitment of practitioners and their assignment to tasks is then accomplished by matching practitioner capabilities with the capabilities required by tasks, while also ensuring that the requisite competency levels are satisfied. Such detailed specification should minimize any potential anxiety of team members due to role blurring [17]. It also allows for a situation where a given task can be assigned to practitioner with other specialty who possess capabilities required by this task. Finally, to ensure the continuity of care within a workflow, we introduce a concept of *task collections* as sets of workflow tasks that should be executed preferrably by the same practitioner.

3.2. Domain model

3.2.1. IHT ontology

The IHT ontology for the extended MET4 system is based on the ontology described in [9,8] and it introduces new and revised concepts and relations that correspond to the specifications outlined in Section 3.1. The ontology is divided into three areas with workflow-, team- and patient-specific concepts respectively. Some of these areas overlap and share concepts. Selected concepts and relations from the ontology that are directly needed for representing the IHT dynamics are given in

Figure 2, where shared concepts and concepts belonging to specific areas are marked with different shades of gray.

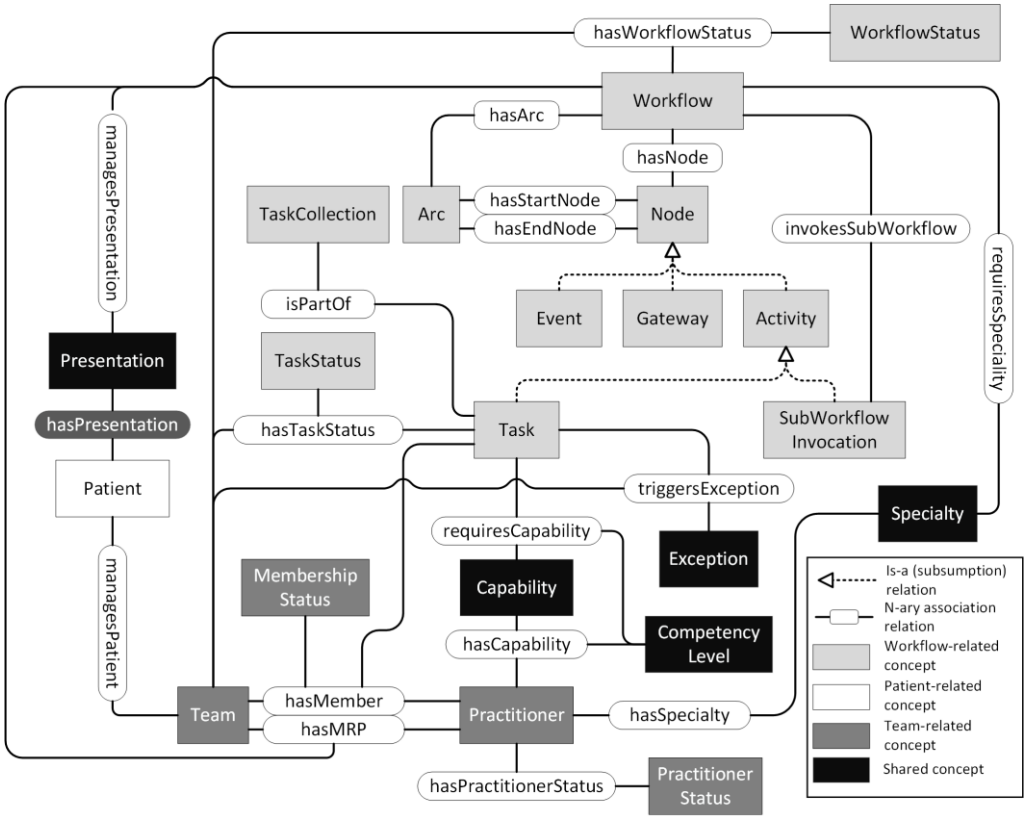

Figure 2 – Selected parts from the IHT ontology

Most of the concepts and relations from the overview in Figure 1 also appear in the IHT ontology in Figure 2, although some relation names are changed to ensure their uniqueness, as required by FOL. For example, the *workflow* concept is associated with required practitioner's *specialty* through the *requiresSpecialty*(*Workflow*, *Specialty*) relation, and the *task* concept is associated with the task collection through the *isPartOf*(*Task*, *TaskCollection*) relation. Two concepts from the conceptual model – *MRP* and *regular member* – have been translated to relations in the IHT ontology for a more concise representation. The *MRP* concept is captured by the *hasMRP(Team, Practitioner, Workflow)* relation, indicating that a specific practitioner is the MRP for a specific team executing a specific workflow. Moreover, the concept of *regular member* is captured by the *hasMember*(*Team*, *Practitioner*, *Task*, *MembershipStatus*) that indicates a specific practitioner is or should be recruited as a regular member to a specific team and assigned a specific task, while also giving the status of the membership. Possible statuses are as follows: *Assigned* – indicating that a practitioner is the member assigned to a task, *Preferred* – indicating that a practitioner is the preferred member to be assigned to a future task (this status is used to ensure that all tasks from a collection are assigned to the same team member), and *Awaiting* – indicating that a practitioner who is busy with some other task should be assigned to the task as soon as she becomes available

The IHT ontology introduces two ternary relations, *hasWorkflowStatus*(*Workflow*, *WorkflowStatus*, *Team*) and *hasTaskStatus*(*Task*, *TaskStatus*, *Team*), which respectively indicate the status of a specific workflow and the status of a specific task for a specific team. Both these relations include a team to account for situations where multiple teams follow the same workflow. Possible workflow statuses are the following: *Running* – a worklow is being executed, *MRPSelection* – a workflow is waiting for MRP selection, and *Completed* – a workflow has been completed. We also distinguish between the following task statuses: *Waiting* – a task is waiting for execution, *MemberSelection* – a task is waiting for member selection and assignment, *Executing* – a task is being executed, and *Executed* – a task has been executed. The ontology captures the status of a practioner through the *hasPractitionerStatus*(*Practitioner*, *PractitionerStatus*) relation. Possible statuses are: *Available* – a practitioner is available and can be assigned a task, *Busy* – a practitioner is currently busy executing some other task, and *NotAvailable* – a pracitioner is not available (i.e., off-shift).

Finally, the IHT ontology introduces the ternary *hasCapability*(*Practitioner*, *Capability*, *CompetencyLevel*) relation to associate a practitioner with a possessed capability and a competency level, and the *requiresCapability*(*Task*, *Capability*, *CompetencyLevel*) relation to associate a workflow task with a required capability and a competency level. While this is different from an overview illustrated in Figure 1, where multiple binary relations were used to capture the same information, such relations are necessary to allow for an efficient representation of capabilities in the instance base.

Finally, we should note that there are certain relations in Figure 1 that do not have their counterparts in the IHT ontology. Specifically, this is the *can be transferred to* relation between MRP and practitioner. This relation is established temporarily during execution time, when a sub-workflow is being invoked, and once a new MRP is appointed it is no longer valid. Thus, given its dynamic and temporal nature, it is realized indirectly through a set of behavioral rules discussed in Section 3.2.2.

3.2.2 Behavioral rules

Behavioral rules describe the dynamics of an IHT. They were developed with the help of clinicians and are formulated using concepts and relations from the IHT ontology. During a reasoning process rules are applied to instances (facts) recorded in the instance base in order to infer the required behavior of an IHT. New facts constructed during the reasoning are stored in the instance base to inform the team dynamics in the future. The application of the behavioral rules is discussed in detail in Section 4 with the help of an illustrative clinical scenario.

The number of behavioral rules is large and because of space limitations we focus here on the rules that concern the new features in the extended MET4 system, specifically: (1) the selection and maintenance of the MRP, (2) the capability-based assignment of practitioners to tasks (both single ones and belonging to a collection), and (3) the identification of exceptions caused by the limited availability of practitioners. For each of these situations, we briefly present the strategy and illustrate it with selected rules.

Selection and maintenance of the MRP

The MRP for a team is selected from available practitioners having the specialty required for managing a disease (rule 1). In the case of an already existing IHT, preference is given to a practitioner who was already associated with that team in the past (rule 1); otherwise any available practitioner with the required specialty will be given the MRP role (rule 2). As already stated, these two rules indirectly implement the *can be transferred to* relation indicated in Figure 1.

Rule 1: $\forall w, s, tm, pr \Big(Workflow(w) \wedge Speciality(s) \wedge Team(tm) \wedge Practitioner(pr) \wedge requiresSpeciality(w, s)$
$\wedge\, hasSpeciality(pr, s) \wedge hasPractitionerStatus(pr, Available)$
$\wedge\, hasWorkflowStatus(w, MRPSelection, tm) \wedge \big(\exists w_1, Workflow(w_1) \wedge hasMRP(tm, pr, w_1)\big)$
$\wedge\, \big(\nexists pr_1, Practitioner(pr_1) \wedge hasMRP(tm, pr_1, w)\big)\Big) \longrightarrow hasMRP(tm, pr, w).$

Rule 2: $\forall w, s, tm, pr, \Big(Workflow(w) \wedge Specialty(s) \wedge Team(tm) \wedge Practitioner(pr) \wedge requiresSpecialty(w, s)$
$\wedge\, hasSpecialty(pr, s) \wedge hasPractitionerStatus(pr, Available)$
$\wedge\, hasWorkflowStatus(w, MRPSelection, tm)$
$\wedge\, \Big(\forall w_1, pr_1, \big(Workflow(w_1) \wedge Practitioner(pr_1) \wedge hasMRP(tm, pr_1, w_1)\big)$
$\rightarrow \neg\big(hasSpecialty(pr_1, s) \wedge hasPractitionerStatus(pr_1, Available)\big)\Big)$
$\wedge\, \big(\nexists p_1, Practitioner(pr_1) \wedge hasMRP(tm, pr_1, w)\big)\Big) \longrightarrow hasMRP(tm, pr, w)$

Capability-based assignment of practitioners to tasks

The assignment of a practitioner to a task follows a capability-based principle. The first choice is to select a practitioner who is preferred for a task (i.e., the task belongs to a collection and the preferred practitioner has already executed some task(s) from the collection) (rule 3). The membership status of a preferred practitioner is established by another behavioral rule (not presented here) that is triggered on assigning the first task from a collection. Alternatively, if a preferred practitioner is not identified or not available, then the task can be assigned to the MRP providing she is capable of executing this task (rule 4). Finally, the assignment may go to any other practitioner who is capable and not busy executing other tasks.

Rule 3: $\forall\, t, tm, pr, \big(Task(t) \wedge Team(tm) \wedge Practitioner(pr) \wedge hasTaskStatus(t, MemberSelection, tm)$
$\wedge\, hasMember(tm, pr, t, Preferred) \wedge hasPractitionerStatus(pr, Available)\big)$
$\longrightarrow hasMember(tm, pr, t, Assigned).$

Rule 4: $\forall t, tm, pr, w, c, cl, cl_1, \big(Task(t) \wedge Team(tm) \wedge Practitioner(pr) \wedge Workflow(w) \wedge Capability(c)$
$\wedge\, hasTaskStatus(t, MemberSelection, tm) \wedge requiresCapability(t, c, cl)$
$\wedge\, hasCapability(pr, c, cl_1) \wedge hasPractitionerStatus(pr, Available) \wedge (cl_1 \geq cl)$
$\wedge\, hasMRP(tm, pr, w)$
$\wedge\, \big(\nexists pr_1, Practitioner(pr_1) \wedge hasMember(tm, pr_1, t, Preferred)$
$\wedge\, hasPractitionerStatus(pr_1, Available)\big)\big) \longrightarrow hasMember(tm, pr, t, Assigned).$

Identification of workflow exception

There are situations when it is not possible to find a practitioner for task execution and we label them as an *exception* (rule 5). If it is possible to identify a practitioner who is currently busy but otherwise capable to execute a task causing the exception, this practitioner is identified as waiting for task assignment (in a sense resolving the exception) (rule 6). Otherwise workflow execution is halted and a decision (outside workflow execution) by the MRP is required.

Rule 5: $\forall t, tm, c, cl, \big(Task(t) \wedge Team(tm) \wedge Capability(c) \wedge hasTaskStatus(t, MemberSelection, tm)$
$\wedge\, requiredCapability(t, c, cl) \wedge \big(\nexists pr, Practitioner(pr) \wedge hasMember(tm, pr, t, Awaited)\big)$
$\wedge\, \big(\nexists pr, Practitioner(pr) \wedge hasPractitionerStatus(pr, Available) \wedge hasCapability(pr, c, cl_1)$
$\wedge\, (cl_1 \geq cl)\big)\big) \longrightarrow \exists x, \big(Exception(x) \wedge triggersException(t, x, tm)\big).$

Rule 6: $\forall x, t, tm, c, pr, cl, cl_1, \big(Exception(x) \wedge Task(t) \wedge Team(tm) \wedge Capability(c) \wedge Practitioner(pr)$
$\wedge\, triggersException(t, x, tm) \wedge hasTaskStatus(t, MemberSelection, tm)$
$\wedge\, requiresCapability(t, c, cl) \wedge hasPractitionerStatus(pr, Busy)$
$\wedge\, hasCapability(pr, c, cl_1) \wedge (cl_1 \geq cl)$
$\wedge\, (\nexists pr_1, Practitioner(pr_1) \wedge hasMember(tm, pr_1, t, Awaited))\big)$
$\longrightarrow hasMember(tm, pr, t, Awaited).$

3.3. Agent class model

The agent class model is presented in Figure 3. It has been limited to the elements that are relevant to the concepts described in the paper. Protocols (identified with solid arrows) have been labeled as p1, p2, etc. – we will refer to these labels in Section 4. The agent class model includes:

- *Semantic repository* that stores the domain model and the instance base with persistent and temporal information characterizing workflows, teams, practitioners and patients. Persistent information includes, among others, definitions of workflows while temporal information includes for example, information about the status of a task.
- *Reasoner* that is a solver (theorem prover/model finder). It operates on the semantic repository (specifically it applies behavioral rules to the instance base) to generate a solution (a "model" in the FOL terminology) that identifies MRP or task-practitioner assignments. This solution is further processed by agent components described below.
- *Electronic health record* (*EHR*) that stores institution-wide patient data.
- *Data synchronizer* agent class that acts as an interface between MET4 and the EHR and supports bidirectional data synchronization,
- *Data manager* agent class that manages data stored in the instance base and responds to queries from other agents. It monitors the instance base and informs other agents about the modifications (e.g., practitioner's status updates).
- *Team manager* agent class that creates and maintains a team (MRP selection, task-practitioner assignment), it interacts with the *reasoner* and interprets solutions.
- *Workflow executor* agent class that controls execution of workflows; it receives information from the *team manager* regarding practitioner to be associated with a task.
- *Practitioner assistant* agent class that operates on mobile devices and interacts through a dedicated interface with the practitioner so she can complete tasks requested by the *workflow executor*.

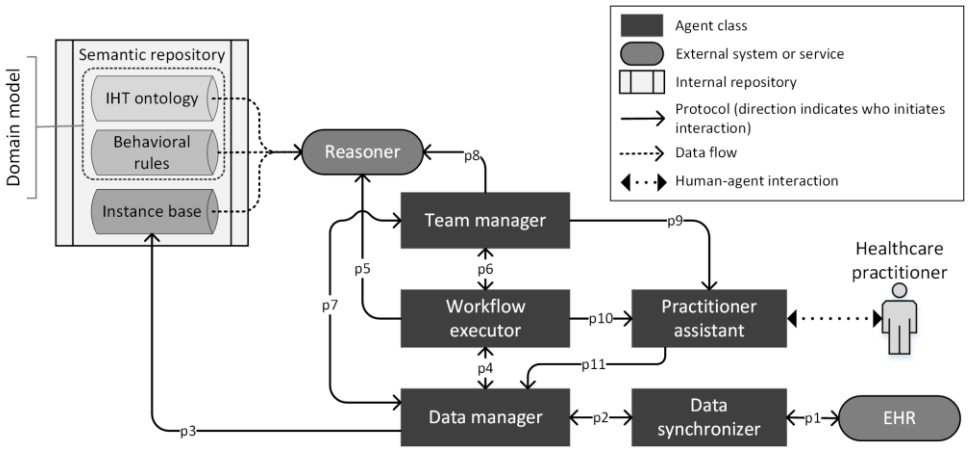

Figure 3 – Agent class model for the extended MET4

The extended MET4 system has more agent classes that provide support for treatment planning, therapy development, and provision of medical evidence as described in [32].

3.4. Proof-of-concept implementation

We implemented the extended MET4 system using the *Workflows and Agent Development Environment* (WADE) [33]. WADE allows for developing, deploying and running MAS, where agents coordinate their activities according to a set of workflow tasks. The environment provides Java programming libraries, visual workflow development tools and an execution environment acting as a middleware for the systems like MET4.

We used the Z3 solver [34] as a *reasoner* because of its extensive API, including Java bindings that facilitate the implementation of the interface between Z3 and the *team manager* agent. The *IHT ontology* and *behavioral rules* are encoded in FOL using the SMT-LIB standard and imported to the *instance base* using the Z3 API.

We evaluated this proof-of-concept implementation using simulated scenarios with multiple teams managing patients according to multiple workflows. We did not observe any deterioration of the system's performance – in particular we did not experienced any meaningful delays caused by frequent invocation of Z3 (in fact, Z3 is one of the most powerful solvers available [35]). These simulations also involved testing the integration of MET4 with an EHR. We used the OpenMRS system [36] as the EHR storing data of fictitious patients and the communication between the *data synchronizer* agent and OpenMRS was realized through REST services.

4. Illustrative example

4.1. Chronic kidney disease

Chronic kidney disease (CKD) involves abnormal kidney function that often co-exists with chronic conditions such as diabetes. Management of advanced CKD involves an IHT composed of a nephrologist (acting as MRP), a dietician, and a nurse. In case of CKD patients who have an elevated risk of cardiovascular (CV) disease, a CV sub-workflow needs to be followed where patient management is temporarily transferred to a cardiologist who becomes the MRP for that sub-workflow. A simplified version of a workflow describing CKD assessment and management developed according to [37] is presented in Figure 4. In this figure, the CKD workflow is labeled as *w_CKD* and the CV risk management sub-workflow as *w_CV_risk_management*. These labels are later used as names of appropriate instances in the *instance base*. The same applies to the labels associated with tasks, sub-workflow invocations, capabilities and specialties.

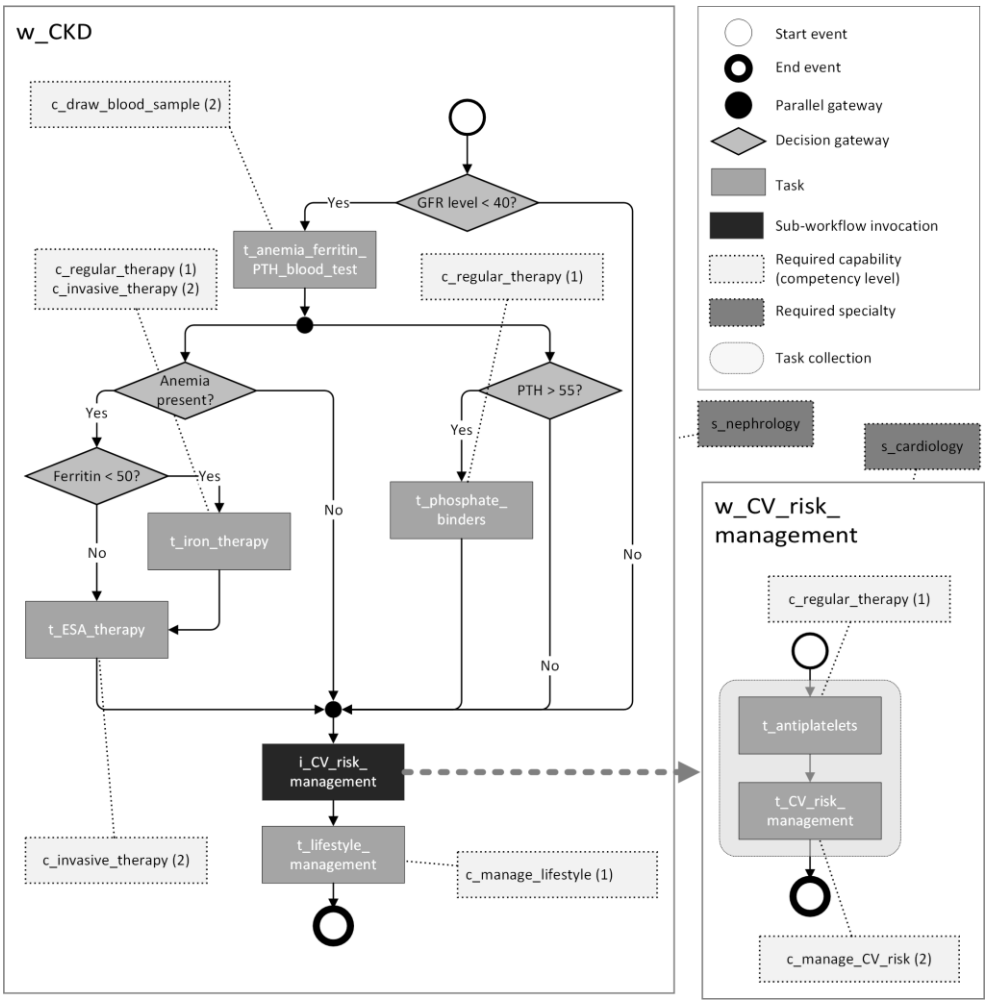

Figure 4 – Workflow for managing CKD

In our example there are 5 practitioners who can participate in a team managing a CKD patient. According to the assumptions and requirements, these practitioners need to be described in terms of their specialties, capabilities, and competency levels (see an overview in Figure 1). This information is represented in Table 1. For simplicity, we assume that competency levels are represented using numerical scale where 1 corresponds to novice, 2 to regular clinical staff member, and 3 to an expert clinician. Each practitioner has an instance of *practitioner assistant* agent class (i.e., there is an instance of *practitioner assistant* for a practitioner named MK).

Table 1 – Considered practitioners

Practitioner	Specialty	Capability and competency level
MK	s_nephrology	c_invasive_therapy (3) c_draw_blood_sample (2)
DR	s_cardiology	c_manage_CV_risk (3) c_regular_therapy (2) c_invasive_therapy (2)
WM	s_dietetics	c_manage_lifestyle (3)
SW	s_nursing	c_regular_therapy (1) c_invasive_therapy (2) c_draw_blood_sample (3)
CK	s_nursing	c_manage_lifestyle (1) c_invasive_therapy (2) c_draw_blood_sample (2)

The information presented in Table 1 is then translated into facts stored in the *instance base* and Figure 5 illustrates the content of the instance base developed from Table 1. For example, for the first row in the table that describes practitioner *MK*, associated facts are stored as entries f2 – f7 in

Figure 5. Specifically, f1 defines an instance of the practitioner concept, f2 defines an instance of the specialty concept, and f3 - f4 define two instances of the capability concept. Furthermore, f5 associates the practitioner instance with the specialty instance, and f6 - f7 associate the practitioner instance with capability instances and competency levels.

f1	Practitioner(MK).
f2	Specialty(s_nephrology).
f3	Capability(c_invasive_therapy).
f4	Capability(c_draw_blood_sample).
f5	hasSpecialty(MK, s_nephrology).
f6	hasCapability(MK, c_invasive_therapy, 3).
f7	hasCapability(MK, c_draw_blood_sample, 2).
f8	Presentation(p_CKD).
f9	Workflow(w_CKD).
f10	managesPresentation(w_CKD, p_CKD).
f11	requiresSpecialty(w_CKD, s_nephrology).
f12	Patient(Jean).
f13	hasPresentation(Jean, p_CKD).
f14	Team(tm_Jean).
f15	managesPatient(tm_Jean, Jean).
f16	hasWorkflowStatus(w_CKD, MRPSelection, tm_Jean).
f17	hasPractitionerStatus(MK, Available).
f18	hasMRP(tm_Jean, MK, w_CKD).
f19	Task(t_anemia_ferritin_PTH_blood_test).
f20	requiresCapability(t_anemia_ferritin_PTH_blood_test, c_draw_blood_sample, 2).
f21	hasTaskStatus(t_anemia_ferritin_PTH_blood_test, MemberSelection, tm_Jean).
f22	hasMember(tm_Jean, MK, t_anemia_ferritin_PTH_blood_test, Assigned).
f23	hasTaskStatus(t_anemia_ferritin_PTH_blood_test, Executing, tm_Jean).
f24	hasPractitionerStatus(MK, Busy).
f25	hasMember(tm_Jean, SW, t_iron_therapy, Awaited).
f26	hasMember(tm_Jean, SW, t_iron_therapy, Assigned).
f27	hasMRP(tm_Jean, DR, w_CV_risk_management).

Figure 5 – Sample facts in the instance base

The *instance base* also stores the definition of the CKD workflow (facts f8 – f11) and the CV risk management sub-workflow. For the sake of simplicity, we do not show complete definitions of workflows such as nodes, arcs, etc.

When a CKD patient (*Jean*) is admitted to a hospital, the *data synchronizer* agent is informed by the OpenMRS EHR about this admission (protocol p1 in Figure 3) and sends a message to the *data manager* agent (protocol p2) that updates the *instance base* (protocol p3) with the facts f12 and f13.

4.2. Selecting the MRP

The *data manager* agent informs the *workflow executor* agent about *Jean*'s admission (protocol p4). In response, the *workflow executor* agent invokes the *reasoner* (protocol p5) to identify a presentation-specific workflow (*w_CKD*) by matching facts f10 and f13, and then requests the *team manager* agent (protocol p6) to create a team to manage *Jean*. The *team manager* agent creates a team (*tm_Jean*), requests the *data manager* agent to update the *instance base* (protocols p7 and p3) with facts f14 and f15, and reports the team creation back to the *workflow executor* agent. The team is initially "empty" (i.e., it has no members).

Before the *w_CKD* workflow can be executed, the MRP for this workflow and the *tm_Jean* team need to be selected. In order to do so, the *workflow executor* agent sends a request to the *team manager* agent (p6) that updates the *instance base* via the *data manager* agent (protocols p7 and p3) with fact f16 indicating that *w_CKD* is waiting for MRP selection. Then the *team manager* agent invokes the *reasoner* (protocol p8) that will use all the facts from the instance base and all

behavioral rules to derive at the MRP selection. At this point let us assume that *MK* is available, as is indicated by fact f17. Considering all the facts and the rules, the *reasoner* uses rule 2 (see Section 3.2.2) to generate a solution, where *MK* is assigned the MRP role in team *tm_Jean* executing the *w_CKD* workflow. This solution is reported to the *team manager* agent that records a new fact f18 in the *instance base* using the *data manager* agent (protocols p7 and p3).

The *team manager* agent also sends a message to the *practitioner assistant* agent (protocol p9) associated with *MK* to inform it about MK being selected as MRP, and finally reports back the selection to the *workflow executor* agent which then starts executing the *w_CKD* workflow shown in Figure 4.

4.3. Assigning a practitioner to a task and executing the task

The GFR (glomerular filtration rate) level for *Jean* retrieved by the *workflow executor* agent from the EHR through *data manager* and *data synchronizer* agents (protocols p4, p2 and p1) is equal to 30, thus the left branch of the workflow, starting at the first decision gateway, is selected, and the *t_anemia_ferritin_PTH_blood_test* task (facts f19 and f20) is to be executed.

The *workflow executor* agent sends a request to the *team manager* agent to select a practitioner for the task at hand (protocol p6). The *team manager* agent sets the status of the *t_anemia_ferritin_PTH_blood_test* task (protocols p7 and p3) to indicate that it is waiting for practitioner assignment (fact f21) and invokes the *reasoner* (protocol p5).

We assume at this point that MK is still available and thus the *reasoner* applies all *behavioral rules* to the *instance base* and uses rule 4 to derive a solution that assigns *MK* to the task to be executed (fact f22). This is because *MK* has the required capability and required competency level (fact f7), and is the MRP for the *tm_Jean* team in the *w_CKD* workflow (fact f18).

The solution is reported to the *team manager* agent, which passes the message back to the *workflow executor* agent. Subsequently, the *workflow executor* agent updates the *instance base* (protocols p4 and p3) to change the status of the task to *Executing* – fact f21 is replaced with f23, and sends a request to the *practitioner assistant* agent associated with *MK* (protocol p10). In response, the *MK*'s *practitioner assistant* agent updates via the *data manager* agent the status of *MK* to *Busy* (protocols p11 and p3) by replacing fact f17 by f24, interacts with *MK* to complete this task, and once it has been completed, reverts *MK*'s status back to *Available*. Finally, the *MK*'s *practitioner assistant* agent responds to the *workflow executor* agent about completing the task, so that the next task from the *w_CKD* workflow can be processed.

4.4. Identifying and handling an exception

Let us assume a situation where all five practitioners are busy when the *workflow executor* agent encounters the *t_iron_therapy* task. That agent notifies the *team manager* agent (protocol p6) that sets the status of *t_iron_therapy* as waiting for practitioner assignment (protocols p7 and p3) and invokes the *reasoner* (protocol p8). The *reasoner* applies all the rules to the facts in the instance base, and in this particular situation first uses rule 5 to identify an exception and then uses rule 6 to handle this exception by deriving a solution, where *SW* is identified as a practitioner to be assigned to this task once he becomes available. The solution is reported to the *team manager* agent that updates the *instance base* with fact f25 (protocols p7 and p3).

The *team manager* agent waits until *SW* completes the current task and *SW's* status is updated (all changes in the *instance base* are reported by the *data manager* agent using protocol p7). The *reasoner* is invoked again – this time the derived solution indicates *SW* as a practitioner selected for *t_iron_therapy* (fact f26). The *team manager* agent updates the *instance base* by removing fact f25 and passes the information to the *workflow executor* agent.

4.5. Invoking a sub-workflow and changing MRP

The *workflow executor* agent encounters the *i_CV_risk_management* task, which indicates passing control to the *w_CV_risk_management* sub-workflow. The MRP for this sub-workflow needs to be selected before the *workflow executor* agent can carry on with the execution. Thus, it sends a request to the *team manager* agent (protocol p6) to find an MRP for *tm_Jean* and *w_CV_risk_management*. The MRP for *w_CV_risk_management* workflow needs to be a specialist in cardiology (*s_cardiology* in Table 1) and, as *MK* does not have this specialty, a new MRP needs to be identified. Let us assume that *DR* is available. The *team manager* agent sets the status of the *w_CV_risk_management* process as waiting for MRP selection (protocols p7 and p3)

and invokes the *reasoner* (p8). The *reasoner* uses rule 2 to derive a solution indicating that *DR* is now the MRP. This solution is reported to the *team manager* agent that adds fact f27 to the *instance base* (protocols p7 and p3) and the processing continues.

Here we need to point out that the *instance base* now contains facts f18 and f27, thus we keep track of the MRPs selected for all workflows executed by the *tm_Jean* team. Once the *w_CV_risk_management* sub-workflow has been completed fact f27 will be removed from the *instance base* and *MK* will be re-assigned back to the MRP role for the *w_CKD*.

5. Discussion

The main contribution of this paper is a new approach to modeling and controlling the IHT dynamics that exploits knowledge of required and available capabilities. Our proposal introduces three semantic components (the IHT ontology, behavioral rules and instance base) that govern the selection of the MRP, capability-based assignment of practitioners to the tasks, and handling the exceptions. In that sense, our research differs from the one described in [22], where the organizational rules are used at system *design time* as opposed to *run time*. We also use FOL as a uniform representation for the all the semantic components, which allows us to infer the desired dynamics in a changing context (e.g., changing statues of the workflows, tasks and practitioners). Finally, the behavioral rules provide a scalable representation of team management strategies (e.g., MRP selection or task-practitioner assignment) and can be revised without the need to modify the implemented system. We illustrated the operation of the semantic elements with the proof-of-concept implementation of the extended MET4 system.

Currently, we are working on incorporating the expanded role of the patient in the operations of an IHT, thus responding to challenges of participatory medicine. For future research, we intend to enrich the semantic components so we can provide support for temporal properties, including task durations and shift changes. We also plan to enhance the task assignment strategies so they allow for balancing load among team members, especially when multiple people are eligible to be added to a team. Finally, we are looking to support more complex team dynamics, especially with regards to handling exceptions related to patient state and to re-allocation of practitioners to the tasks.

Acknowledgments

This research was supported by grants from the NSERC CREATE Program in Healthcare Operations and Information Management and the Telfer School of Management Research Fund. We would like to acknowledge contribution of Mr. Runzhuo Li in implementing an earlier version of the MET4 system. We would like to thank the reviewers for their comments and suggestions.

References

1. Jun GT, Ward J, Morris Z, Clarkson J (2009) Health care process modelling: which method when? Int J Qual Health Care 21 (3):214-224
2. Wen J, Schulman KA (2014) Can team-based care improve patient satisfaction? a systematic review of randomized controlled trials. PloS One 9 (7)
3. Oandasan I, D'Amour D, Zwarenstein M, Barker K, Purden M, Beaulieu M, Reeves S, Nasmith L, Bosco C, Ginsburg L (2004) Interdisciplinary education for collaborative, patient-centred practice: research and findings report. Health Canada,
4. Andreatta PB (2010) A typology for health care teams. Health Care Manage Rev 35 (4):345-354
5. Isern D, Sánchez D, Moreno A (2010) Agents applied in health care: A review. Int J Med Inform 79 (3):145-166
6. Isern D, Moreno A, Sánchez D, Hajnal Á, Pedone G, Varga LZ (2011) Agent-based execution of personalised home care treatments. Appl Intell 34 (2):155-180
7. Astaraky D, Wilk S, Michalowski W, Andreev P, Kuziemsky C, Hadjiyannakis S A multiagent system to support an interdisciplinary healthcare team: a case study of clinical obesity management in children. In: Proceedings of the VIII Workshop on Agents Applied in Healthcare and Conference on Artificial Intelligence in Medicine, Murcia, Spain, 2013. pp 69-79
8. Kuziemsky C, Astaraky D, Wilk S, Michalowski W, Andreev P (2014) A framework for incorporating patient preferences to deliver participatory medicine via interdisciplinary healthcare teams. AMIA Annu Symp Proc:835-844
9. Wilk S, Astaraky D, Amyot D, Runzhuo L, Kuziemsky C, Andreev P (2015) MET4: supporting workflow execution for interdisciplinary healthcare teams. In: Fournier F, Mendling J (eds)

Business Process Management Workshops. BPM 2014 International Workshops, Eindhoven, The Netherlands, September 7-8, 2014, Revised Papers. LNBIP, vol 202. Springer, pp 40-52
10. Pavlov V, Schukin A, Cherkasova T (2013) Exploring automated reasoning in first-order logic: tools, techniques and application areas. In: Klinov P, Mouromtsev D (eds) Knowledge Engineering and the Semantic Web. 4th International Conference, KESW 2013, St. Petersburg, Russia, October 7-9, 2013. Proceedings. CCIS, vol 394. Springer, pp 102-116
11. Scheuerlein H, Rauchfuss F, Dittmar Y, Molle R, Lehmann T, Pienkos N, Settmacher U (2012) New methods for clinical pathways-Business Process Modeling Notation (BPMN) and Tangible Business Process Modeling (t.BPM). Langenbecks Arch Surg 397 (5):755-761. doi:10.1007/s00423-012-0914-z
12. Strasser M, Pfeifer F, Helm E, Schuler A, Altmann J (2011) Defining and reconstructing clinical processes based on IHE and BPMN 2.0. Stud Health Technol Inform 169:482-486
13. Peleg M (2013) Computer-interpretable clinical guidelines: a methodological review. J Biomed Inform 46 (4):744-763. doi:10.1016/j.jbi.2013.06.009
14. Müller R, Rogge-Solti A (2011) BPMN for healthcare processes. In: Eichhorn D, Koschmider A, Zhang H (eds) Proceedings of the 3rd Central-European Workshop on Services and their Composition, ZEUS 2011. CEUR-WS.org, pp 65-72
15. Grando A, Peleg M, Glasspool D (2010) A goal-oriented framework for specifying clinical guidelines and handling medical errors. J Biomed Inform 43 (2):287-299
16. Clark PG, Drinka TJK (2000) Health care teamwork: interdisciplinary practice and teaching. Greenwood Publishing Group,
17. Sims S, Hewitt G, Harris R (2015) Evidence of collaboration, pooling of resources, learning and role blurring in interprofessional healthcare teams: a realist synthesis. J Interprof Care 29 (1):20-25
18. Sims S, Hewitt G, Harris R (2015) Evidence of a shared purpose, critical reflection, innovation and leadership in interprofessional healthcare teams: a realist synthesis. J Interprof Care 29 (3):209-215. doi:10.3109/13561820.2014.941459
19. Kuziemsky CE, Varpio L (2011) A model of awareness to enhance our understanding of interprofessional collaborative care delivery and health information system design to support it. Int J Med Inform 80 (8):e150-160
20. Riley W, Lownik E, Parrotta C, Miller K, Stan D (2011) Creating high reliability teams in healthcare through In situ simulation training. Adm Sci 1 (1):14-31
21. Buljac-Samardzic M, Dekker-van Doorn CM, van Wijngaarden JD, van Wijk KP (2010) Interventions to improve team effectiveness: a systematic review. Health Policy 94 (3):183-195
22. Zambonelli F, Jennings NR, Wooldridge M (2001) Organisational rules as an abstraction for the analysis and design of multi-agent systems. Int J Softw Eng Knowl Eng 11 (3):303-328
23. van der Vecht B, Dignum F, Meyer JJC, Dignum V (2009) Organizations and autonomous agents: bottom-up dynamics of coordination mechanisms. In: Coordination, Organizations, Institutions and Norms in Agent Systems IV. COIN 2008 International Workshops, COIN@AAMAS 2008, Estoril, Portugal, May 12, 2008. COIN@AAAI 2008, Chicago, USA, July 14, 2008. Revised Selected Papers. LNAI, vol 5428. Springer, pp 17-32
24. van Riemsdijk MB, Dignum V, Jonker CM, Aldewereld H (2011) Programming role enactment through reflection. In: 2011 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), vol 2. IEEE, pp 133-140
25. Panagiotidi S, Alvarez-Napagao S, Vázquez-Salceda J (2014) Towards the norm-aware agent: bridging the gap between deontic specifications and practical mechanisms for norm monitoring and norm-aware planning. In: Balke T, Dignum F, van Riemsdijk MB, Chopra AK (eds) Coordination, Organizations, Institutions, and Norms in Agent Systems IX. COIN 2013 International Workshops, COIN@AAMAS, St. Paul, MN, USA, May 6, 2013, COIN@PRIMA, Dunedin, New Zealand, December 3, 2013. Revised Selected Papers. LNAI, vol 8386. Springer, pp 346-363
26. Panagiotidi S, Vázquez-Salceda J (2011) Norm-aware planning: semantics and implementation. In: 2011 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), vol 3. IEEE, pp 33-36
27. Weerakkody G, Ray P (2003) CSCW-based system development methodology for health-care information systems. Telemed J E Health 9 (3):273-282
28. Moreno A, Valls A, Riaño D (2005) PalliaSys: agent-based proactive monitoring of palliative patients. In: Proceedings of the IV International Workshop on Practical Applications of Agents and Multi-Agent Systems (IWPAAMS). pp 101-110
29. Su C-J, Wu C-Y (2011) JADE implemented mobile multi-agent based, distributed information platform for pervasive health care monitoring. Appl Soft Comput 11 (1):315-325

30. Isern D, Sánchez D, Moreno A (2007) HeCaSe2: a multi-agent ontology-driven guideline enactment engine. In: Burkhard H-D, Lindemann G, Verbrugge R, Varga LZ (eds) Multi-Agent Systems and Applications V. 5th International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2007, Leipzig, Germany, September 25-27, 2007. Proceedings. LNAI, vol 4696. Springer, pp 322-324
31. Garcia-Ojeda JC, DeLoach SA, Robby, Oyenan WH, Valenzuela J (2008) O-MaSE: a customizable approach to developing multiagent development processes. In: Luck M (ed) Agent-Oriented Software Engineering VIII: the 8th International Workshop on Agent Oriented Software Engineering (AOSE 2007). LNCS, vol 4951. Springer, pp 1-15
32. Wilk S, Michalowski W, O'Sullivan D, Farion K, Sayyad-Shirabad J, Kuziemsky C, Kukawka B (2013) A task-based support architecture for developing point-of-care clinical decision support systems for the emergency department. Methods Inf Med 52 (1):18-32
33. WADE: workflows and agents development environment. http://jade.tilab.com/wade/. Accessed January 10, 2015
34. Z3: a high-performance theorem prover. http://z3.codeplex.com/. Accessed January 10, 2015
35. Cok DR, Stump A, Weber T (2015) The 2013 evaluation of SMT-COMP and SMT-LIB. J Autom Reason 55 (1):61-90
36. Wolfe BA, Mamlin BW, Biondich PG, Fraser HSF, Jazayeri D, Allen C, Miranda J, Tierney WM (2006) The OpenMRS system: collaborating toward an open source EMR for developing countries. AMIA Annu Symp Proc:1146-1146
37. Levin A, Hemmelgarn B, Culleton B, Tobe S, McFarlane P, Ruzicka M, Burns K, Manns B, White C, Madore F, Moist L, Klarenbach S, Barrett B, Foley R, Jindal K, Senior P, Pannu N, Shurraw S, Akbari A, Cohn A, Reslerova M, Deved V, Mendelssohn D, Nesrallah G, Kappel J, Tonelli M (2008) Guidelines for the management of chronic kidney disease. CMAJ 179 (11):1154-1162