An Ontology-driven Framework to Support the Dynamic Formation of an Interdisciplinary Healthcare Team

Szymon Wilk[a,b,*], Mounira Kezadri-Hamiaz[b,1], Daniel Amyot[c], Wojtek Michalowski[b], Craig Kuziemsky[b,2], Nihan Catal[c], Daniela Rosu[b,3], Marc Carrier[d], Randy Giffen[b,e]

[a] Institute of Computing Science, Poznan University of Technology,

Piotrowo 2, 60-965 Poznan, Poland

[b] Telfer School of Management, University of Ottawa,

55 Laurier Ave E, Ottawa, ON K1N 6N5, Canada

[c] School of Electrical Engineering and Computer Science, University of Ottawa,

800 King Edward Avenue, Ottawa, ON K1N 6N5, Canada

[d] The Ottawa Hospital Research Institute, 725 Parkdale Ave, Ottawa, ON K1Y 4E9, Canada

[e] Business Analytics Solutions, IBM, 3600 Steeles Avenue, East Markham, ON L3R 9Z7, Canada

[*] Corresponding author; e-mail: szymon.wilk@cs.put.poznan.pl

[1] Present address: College of the Computer Science and Engineering, Taibah University, University Road, PoBox 344 , 41477 Medina, Saudi Arabia

[2] Present address: MacEwan University, 7-257P, 10070 104 Avenue NW, Edmonton, AB T5J 4S2, Canada

[3] Present address: Center for Social Services Engineering, University of Toronto, 40 St. George Street, Toronto, ON M5S 2E4, Canada

Abstract[+]

Background and purpose: *Teamwork has become a modus operandi in healthcare and delivery of patient care by an interdisciplinary healthcare team (IHT) is now a prevailing modality of care. We argue that a formal and automated support framework is needed for an IHT to properly leverage information technology resources. Such a framework should allow for patient's preferences and expand a representation of clinical workflow with a formal model of dynamic formation of a team, especially with regards to team leader- and membership, and the assignment of the tasks to team members. Our goal was to develop such a support framework, present its prototype software implementation and verify the implementation using a proof-of-concept use case. Specifically, we focused on clinical workflows for in-patient tertiary care and on patient preferences with regards to selecting team members and team leaders.*

Materials and methods: *Drawing on the research on clinical teamwork we defined the conceptual foundations for the proposed framework. Then, we designed its architecture and used ontology-driven design and first-order logic with associated reasoning methods to create and operationalize architectural elements. Finally, we incorporated existing solutions for business workflow modeling and execution as a backend for implementing the proposed framework.*

Results: *We developed a Team and Workflow Management Framework (TWMF) with semantic components that allow for formalizing and operationalizing team formation in in-patient tertiary care setting and support provider-related patient preferences. We also created a prototype software implementation of TWMF using the IBM Business Process Manager platform. This implementation was evaluated in several simulated patient scenarios.*

Conclusions: *TWMF integrates existing workflow technologies and extends them with the capabilities to support dynamic formation of the IHT. Results of this research can be used to support real-time*

[+] Abbreviations: BPMN = Business Process Model and Notation, HIS = hospital information system, IHT = interdisciplinary healthcare team, IT = information technology, MRP = most responsible physician, TWMF = team and workflow management framework

execution of clinical workflows, or to simulate their execution in order to assess the impact of various conditions (e.g., patterns of work shifts, staffing) on IHT operations.

Keywords

Team-based care delivery; interdisciplinary healthcare team; dynamic team formation; clinical workflow; workflow execution engine

Highlights

- Complex patients' care is provided by interdisciplinary healthcare teams (IHTs)
- Dynamic formation of the IHT posits challenges and asks for new support methods
- We propose a novel framework to model and operationalize dynamic IHT formation
- We created proof-of-concept implementation using IBM Business Process Manager

1 Introduction

Biomedical research has made significant strides at developing methods and tools to support care delivery by individual healthcare providers, while, for a number of reasons, teamwork has become a desired care delivery model [1]. On the clinical side, these reasons include the increasing complexity of patients' conditions (e.g., chronic diseases with associated morbidities) combined with increasing medical specialization and fragmentation of disciplinary knowledge. On the care delivery side, it is recognized that the management of complex clinical conditions by an interdisciplinary healthcare team (IHT) that includes clinicians and allied healthcare providers from various disciplines working jointly to achieve a common goal of patient-centered care [2] results in improved health outcomes [3]. For simplicity, in the text we will use the generic term "clinician" to indicate any healthcare provider, e.g., physician, nurse or physiotherapist. In cases where precise distinction of providers is needed, we use a specific term.

Care delivery by an IHT is often defined as a "*complex process in which different types of staff work together to share expertise, knowledge, and skills to impact patient care*" [4]. Such care delivery is difficult to operationalize because of the need to consider providers simultaneously as individuals and as members of an IHT operating in a fluid environment where multiple tasks and teams co-exist and where one team member is usually involved in the care of multiple patients [5-7]. Karsh et al. state that IHT operationalization goes beyond designing a new front-end interface to an existing health information technology (IT) system but rather requires a complete system redesign [8]. As demonstrated in [9], the lack of properly redesigned health IT system may diminish the benefits of team-based patient management and may result in inferior patient outcomes.

Research on teamwork processes and interactions in an IHT has largely focused on *how* to model their certain aspects instead of *what* particular aspect should be modeled. Mabry et al. [10] described the use of multiple specialized Intelligent Monitor Agents collaborating in a team-like environment to carry out monitoring and diagnostic tasks in emergency trauma scenarios. Similarly, Taboada et al. [11] proposed an agent-based modeling and simulation tool to assist the operations of an Emergency Room. The implications of the collaboration of autonomous agents with human actors have been discussed by Grosz

[12]. Huguet et al. [13] described the use of a multi-agent architecture to model and interpret the effects of different types of communication errors in the context of field training for medical teams. Ruan et al. [14] developed an agent-based model that is integrated with a workflow system to improve support for an IHT in palliative care setting. Our earlier research [15] used agent-based modeling to represent selected few aspects of team formation. Similar – to some extent – research also is conducted in team science but there it is more concerned with the education of team members than with the overall operationalization of care delivery by an IHT [2].

Activities of an IHT are usually guided by a clinical workflow that specifies tasks to be completed and define their precedence. Despite extensive research on the formalization and execution of business workflows [16], there is limited application of this research in healthcare, mostly because the execution of a business workflow is task-oriented as opposed to being team-oriented [17]. Existing workflow management systems (e.g., [18] or [19]), although effective for executing individual workflows, are ill-suited for executing workflow designed for an IHT. This issue was addressed by Grando et al. who defined patterns in order to model task assignment and delegation as part of team-based care delivery [20]. Others have extended that research by modeling the tasks that an IHT executes and describing how to allocate these tasks to team members [21]. However, they considered IHT membership to be static and did not account for its dynamic nature.

In our research we focus on a specific aspect of the teamwork – dynamic formation of an IHT. It involves identification and the management of team leadership, team membership, and the assignment of tasks to team members, [4, 22]. Every IHT must have a *leader*, i.e., a clinician who makes definitive decisions related to patient management (hospital admission, care path, discharge, etc.). In most cases, the leader is a physician who takes the role of "most responsible physician" (MRP). Other members of the team execute workflow tasks that are compatible with their skills, roles, and access privileges, and membership in an IHT evolves as members of a team finish their shifts, are called to other patients, or execute new tasks requiring different clinical expertise [23]. There are additional challenges of supporting dynamic formation of the IHT, including blurring of the roles of team members [22], the required continuity of team leadership [24], and a need to account for patient preferences whenever possible. It is clear that more research is needed in order to gain a better understanding of the teamwork processes governing an IHT [25, 26], and how to align dynamic formation of an IHT with patient preference models.

In this paper, we build on and significantly expand upon our earlier work [15, 27, 28] and present an extended framework called the *Team and Workflow Management Framework* (TWMF) that allows modeling and operationalizing the dynamic formation of an IHT while also incorporating patient preferences into clinical workflows. TWMF aims at in-patient tertiary care – a setting that is characterized by high team volatility [25] and therefore asks for better support centered on IHT formation. TWMF currently considers patient preferences for attending clinicians and therefore affecting MRP selection and team membership. However, in our earlier work we discussed more complex preference models that involve treatment options, if available [27], thus, TWMF can be extended to handle such models and such extension is discussed in the last section.

Our design and implementation of the TWMF draws upon frameworks for designing and evaluating health IT such as the context of system testing continuum developed by Kushniruk et al. [29]. The continuum represents system evaluation starting with experimental simulations of representative cases before moving into clinical laboratory simulations, and finally clinical simulations in real care delivery settings. The work presented here takes place at the earlier stages of the system testing continuum where the TWMF and associated proof-of-concept prototype are tested in expanded experimental simulation using comprehensive use cases of care delivery by an IHT.

2 Materials and Methods

2.1 Conceptual Foundations

Drawing on the literature briefly summarized in the previous section, we defined the conceptual foundations for the TWMF assuming that:

- an IHT is formed at a time when patient’s management for a specific presentation or condition starts (either in an Emergency Room or upon transfer to the ward),

- a patient is managed according to a workflow appropriate for a reported presentation or diagnosed condition (we refer to such workflow as *primary* workflow). Primary workflow that is associated with specific presentation(s) describes diagnostic phases with a goal of identifying underlying condition(s). Once the diagnosis has been made, such workflow invokes condition-specific workflows that handle further patient management.
- there is a 1:1 relationship between an IHT and a patient. This implies that one team managing multiple patients is modeled as multiple IHTs co-existing at the same time.

A team must have an MRP that overlooks execution of a workflow and who becomes the first member of a team. This role is assigned to one of the available physicians who have the clinical specialty and expertise appropriate for coordination of the primary workflow being executed. For example, in the case of the workflow for managing a radical prostatectomy patient, the MRP is staff surgeon, while the workflow for managing chronic hypertension may ask for a junior cardiologist as MRP. The MRP executes some tasks and we refer to these tasks as MRP-specific tasks. In practice, these tasks involve making decisions about admitting or discharging a patient, recording progress notes, or ordering laboratory or imaging tests. An additional dimension considered in selecting the MRP involves patient's preferences.

As already mentioned, a workflow may invoke another workflow (we refer to an invoking workflow as a parent and to an invoked workflow as a child) and this triggers a need to identify an MRP for the child workflow. If the child workflow asks for a different clinical specialty than the parent one, then a different physician is recruited for the team. On the other hand, if the execution of child workflow requires the same specialty as the parent one or does not explicitly state the specialty, the MPR of the parent workflow is retained for execution of the child workflow. Once the execution of the child workflow is completed, its MRP is released, and the MRP associated with the parent workflow is restored. It is also possible that the physician who plays the MRP role becomes unavailable, and then a different available physician, with the required specialty and experience, is selected for the MRP role.

An IHT exists as long as the patient is managed for the identified condition. Members of the team are recruited dynamically from among available clinicians, as dictated by the execution of tasks from the workflow. After the task is completed, the assigned clinician is released and can be recruited again to the same or to a different IHT. Past members of an IHT are given priority when considered for assignment to new tasks in order to maintain stability in team formation and continuity with/for the patient. A similar approach is followed with the MRP, who in addition to MRP-specific tasks is given priority to execute other tasks. We also assume that continuity of care may require that several tasks should be assigned to the same clinician. These tasks are grouped into a *collection* and the clinician who is capable of executing all tasks in a collection is considered as the first choice for the assignment.

While relying on clinical specialty is sufficient for selecting MRP, this is too restrictive for tasks that require capability-based assignment of clinicians. IHTs can be characterized by role variability [23] where assignment of clinicians to tasks is driven not by their *specialty* but by their clinical *capabilities* (i.e., ability to conduct a test or clinical procedure). We assume that a clinician may possess a number of capabilities, each annotated with a level of clinical expertise (i.e., clinical expertise of staff physician is more extensive – has higher level – than this of a senior resident). A task is ascribed with capabilities and levels of expertise needed for its execution. A clinician is capable of executing a task if her/his expertise matches the minimum expertise level required by the task. The distinction between role-based and capability-based assignment of a clinician to a task is similar to the distinction between the role-based and attribute-based access control considered in computer systems, with the latter being more flexible and fine-grained [30]. We assume that MRP-specific tasks do not need detailed characterization in terms of required capabilities as the physician playing the role of MRP is by definition capable of executing MRP-specific tasks. Since a specialty-based verification is conducted when the MRP is selected, no additional checking during assignment of MRP-specific tasks is required.

Finally, we distinguish between urgent and normal tasks. The former category is handled in a special manner – if a clinician capable of executing an urgent task is currently assigned to a normal task, and no other capable clinician is available, then the execution of the normal task is interrupted and this clinician is reassigned to the urgent task. In the exceptional circumstances when there are no available clinicians assignable to an urgent task, the MRP is informed and the workflow execution is halted. The MRP needs to decide and identify manually the required clinician.

2.2 Ontology-driven Design

We derived the architecture of TWMF by applying ontology-driven design – a paradigm for constructing information systems that can manage dynamic domain models and heterogeneous knowledge [31]. In comparison to approaches with an underlying database, ontology-driven design allows for having expressive domain models that capture complex relationships between concepts (e.g., compositions of relations) and impose complex constraints on these concepts (e.g., related to cardinality). From the architectural perspective, in addition to the *application* and *business logic layers*, ontology-driven design requires creating the *ontology layer* and the *ontology access layer* that executes ontological queries requested by the application logic layer. An interesting example of a system developed following this paradigm is Insight – a platform for integrating and analyzing data for self-management of epilepsy [32].

2.3 Logical Reasoning

We used first order logic (FOL) [33], also called predicate logic, to represent the knowledge describing dynamic behaviour of an IHT and to reason from it. We chose FOL over Web Ontology Language (OWL), typically used to represent ontologies [31], due to better expressiveness of FOL and more advances reasoning abilities. Specifically, FOL operates on theories (sets of logical statements describing domain knowledge) that specify concepts and relations, define rules that infer these relations, and establish facts representing instances of concepts and their relations. Specialized reasoning techniques allow checking for statements (elements of a description of team's dynamic formation) that can be derived (entailed) from a theory.

3 Results

3.1 TWMF Architecture

The architecture of the TWMF together with a *workflow execution engine* (*execution engine* for short) and *hospital information system* (HIS) is shown in Figure 1. The execution engine is responsible for executing workflows represented with process-oriented notations while the HIS (which can actually be a group of systems) manages patient data and notifies other systems about relevant events, such as patient admission. The TWMF encapsulates the need to model both the structure and the dynamic formation of an IHT. This is accomplished by the TWMF having the following elements:

- *semantic components* – an *IHT ontology* that defines concepts and relations describing the structural components associated with an IHT team including clinicians, patients, and elements of the workflow, *formation rules* that model the dynamic formation of the IHT, and a *fact base* that contains facts representing instances of concepts from the IHT ontology and relations between these instances (i.e., patient is a concept in the ontology and “John Doe” is an instance of this concept recorded as a fact),
- a *reasoner* that uses the semantic components as a reasoning context to infer the dynamic formation of the team (MRP selection and assignment of tasks to clinicians) and to derive instructions for the execution engine,
- a *team and workflow controller* (*controller* in short) that operationalizes the framework and interacts with external systems. It is responsible for communicating with the HIS, the execution engine, interpreting reasoner’s responses, and updating the fact base.

As per the ontology-driven design, the semantic components form the ontology layer, the controller constitutes the ontology access and the application logic layers, and the reasoner encapsulates the business logic layer. We should also note that the boundary between the business logic and ontology layers is not sharp, as the reasoner relies on the formation rules from the ontology layer to operationalize the business logic (i.e., management of the IHT dynamic formation). Figure 1 illustrates the main interactions between the framework components and the external systems (e.g., a HIS). Specifically, HIS handles and manages all patient data while the controller manages interactions and acts as a bridge between external systems and the other TWMF components. We further discuss TWMF interactions as well as specific operations of selected TWMF components in greater detail in Appendix 3.

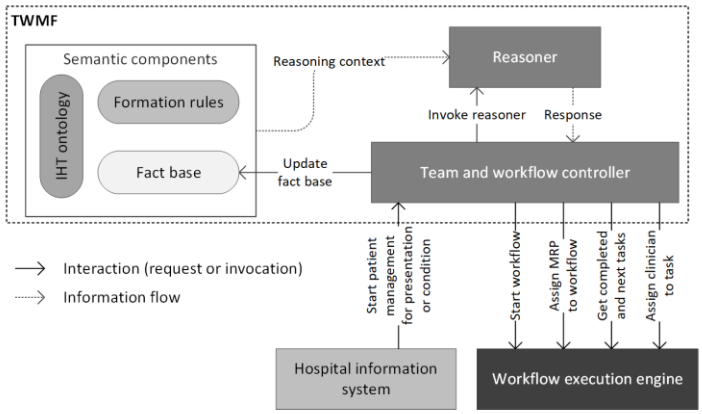

Figure 1. Architecture of TWMF with HIS and execution engine

3.2 *IHT Ontology*

An ontology is usually defined as "an explicit specification of a conceptualization" [34] and its purpose is to represent entities (concepts) and relations between them. We use FOL as ontology a representation language that formalizes the semantic components and for derivation, from formation rules discussed later in the text, of statements describing the dynamic formation of an IHT.

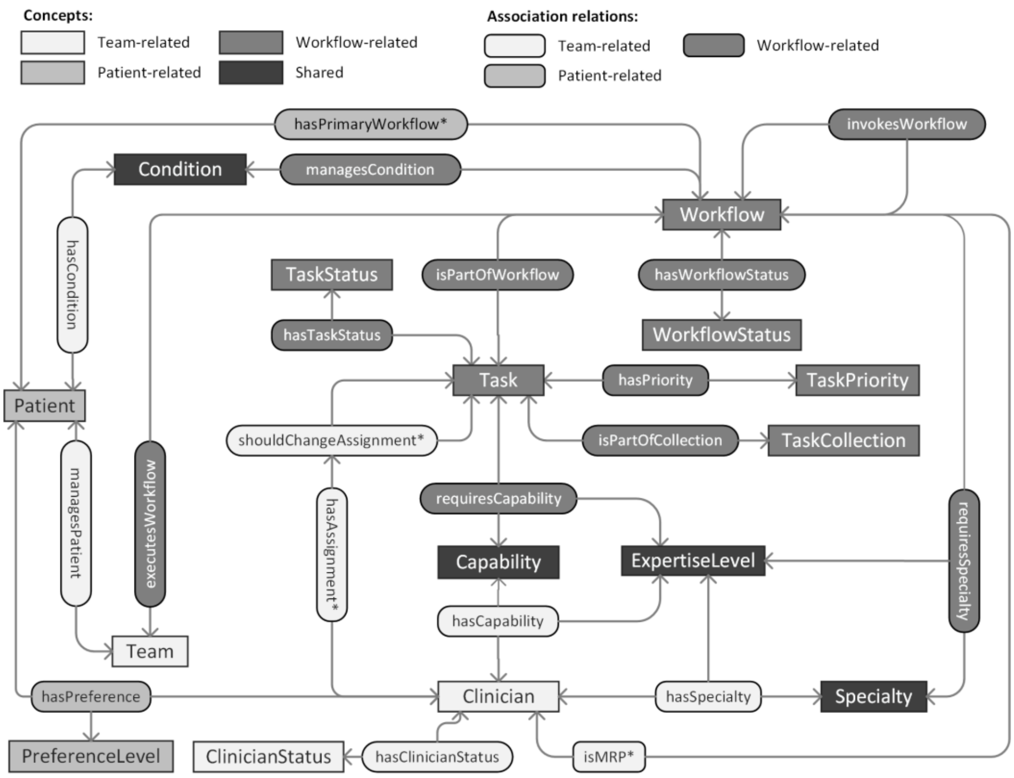

Figure 2. IHT ontology

The IHT ontology, presented in Figure 2, is compatible with the business workflow model processed by the execution engine and it defines concepts and relations required to represent the structure of an IHT and the clinical workflow. Following a postulate that team-oriented workflow representation should be created around the patient [17], the IHT ontology also defines patient-related concepts. We note here that several association relations do not need to be defined explicitly but rather are established automatically by the facilitating formation rules (see Section 3.3.1). We refer to these relations as "inferred" relations and mark them in Figure 2 with "*".

The ontology is partitioned into three sub-ontologies, each containing concepts and relations pertinent to workflow, patient, and a team. Description of the sub-ontologies with the concepts and relations is provided in Appendix 1.

3.3 Formation Rules

In this section, we discuss the formation rules that model the dynamic formation of an IHT. While the structure of formation rules is driven by the IHT ontology and the FOL principles, for the sake of readability in Table 1 we present rules in a plain language. A formal representation of these rules is given in Appendix 2. It is important to note that the ontology and the formation rules can be easily customized to a particular setting, policies, and different dynamic composition patterns.

Table 1. Selected formation rules

r1:	**if** patient *pt* has condition *co* **and** workflow *w* manages condition *co*, **then** mark workflow *w* as primary for patient *pt*
r2:	**if** clinician *cl* is (or has been) MRP for workflow *w* executed by team *tm*, **then** mark clinician *cl* as past or current member of team *tm*
r3:	**if** clinician *cl* is (or has been) assigned to task *t* that is part of workflow *w* executed by team *tm*, **then** mark clinician *cl* as past or current member of team *tm*
r4:	**if** clinician *cl* has all capabilities required by task *t* with sufficient expertise levels, **then** mark clinician *cl* as capable of task *t*
r5:	**if** clinician *cl* has a specialty required by workflow *w* with sufficient expertise level, **then** mark clinician *cl* as eligible for workflow *w*
r6:	**if** team *tm* executes workflow *w* **and** clinician *cl* is eligible for workflow *w* **and** clinician *cl* is on shift **and** clinician *cl* is past or current member of team *tm*, **then** mark clinician *cl* as candidate for MRP for workflow *w*
r8:	**if** workflow *w* does not specify a required specialty **and** workflow *w* is invoked by parent workflow w_p **and** clinician *cl* is MRP for workflow w_p, **then** mark clinician *cl* as candidate for MRP for workflow *w*

r9:	**if** task *t* is MRP-specific **and** task *t* is part of workflow *w* **and** clinician *cl* is MRP for workflow *w*, **then** mark clinician *cl* as candidate for assignment to task *t*
r10:	**if** task *t* is part of workflow *w*, **and** task *t* is not part of any task collection **and** task *t* is not MRP-specific **and** clinician *cl* is capable of task *t* **and** clinician *cl* is free **and** clinician *cl* is past or current member of team *tm* executing workflow *w*, **then** mark clinician *cl* as candidate for assignment to task *t*
r12:	**if** task *t* belongs to task collection *tc* **and** clinician *cl* has been already assigned to at least one task from task collection *tc* **and** clinician *cl* is capable of all not yet executed tasks from task collection *tc* **and** clinician *cl* is free, **then** mark clinician *cl* as candidate for assignment to task *t*
r16:	**if** task t_u has urgent priority **and** no clinician capable of executing task t_u is free **and** clinician *cl* capable of executing task t_u is assigned to and executing task t_n with normal priority, **then** mark tasks t_n and t_u for re-assignment

3.3.1 Facilitating Rules

These rules are used to establish *inferred* relations based on facts stored in the fact base. Rule r1 (for this rule and all subsequent ones see Table 1 and Appendix 2 for their FOL-based formalization) assigns a primary workflow to a patient. Rules r2 and r3 identify past and current members of a team. Rule r4 tags a clinician as capable of executing a task, so she/he may become a candidate for assignment to this task. Finally, rule r5 tags a clinician as potential MRP for the workflow.

3.3.2 Rules for Selecting the MRP for a Workflow

Rule r6 stipulates that a clinician is on a shift, however, she/he may be currently busy with other task because MRP-specific tasks are of administrative nature and may be delayed. Rule 7, omitted from Table 1, is similar to r6, but it drops the requirement of team membership for a clinician. Rule r8 models a situation where a workflow does not explicitly state the required specialty for the MRP – this may happen with the execution of a child workflow. In such a case, the MRP associated with the parent workflow becomes the eligible candidate.

Rules for selecting the MRP (and other rules assigning clinicians to the tasks) may identify multiple suitable candidates. In such a situation, the final selection is made automatically by the controller, which considers patient's preferences specified using the *hasPreference* relation and selects the most preferred care provider. If there is still a tie, then the controller makes the choice randomly from the eligible candidates.

Our current strategy of MRP selection treats patient preferences as a secondary factor in relation to continuity of care as shown in rule r6 – past or current team members are considered first, even though they may be other clinicians who are preferred by the patient. It is possible to make patient preferences more important by incorporating them directly into the MRP selection rules.

3.3.3 Rules for Assigning a Clinician to a Task

When assigning a clinician to a task, we need to consider the following three situations, depending on whether the current task: (1) is MRP-specific, (2) is not MRP-specific and does not belong to a task collection, and (3) is not MRP-specific and belongs to a task collection.

The first situation is the easiest to model and is managed by rule r9. Here we need to recall that it is assumed that MRP-specific tasks are mostly of administrative nature, have normal priority level, and their execution may be delayed when the MRP is busy.

The second situation is more complex and it is managed by two rules – r10 and r11. Rule r11 is similar to r10 – it only drops the requirement of past/current team membership, so it is not included in Table 1.

The third situation requires the sequence of four rules – r12 to r15. Rule r12 is the most comprehensive in terms of requirements imposed on candidates for assignment. Rule r13 drops the requirement for being assigned to at least one earlier task from the collection, yet it still requires the clinician to be past or current team member. Rule r14 drops the requirement for team membership, and rule r15 no longer requires the clinician to be capable of executing all remaining tasks in the collection. Since rules r13 – r15 are simplified versions of rule r12, they are not presented in Table 1.

It is possible that no clinician is available to be assigned to a task. If this is a task with normal priority level, then its execution can be postponed until a suitable clinician becomes available. However, if the task is urgent, a clinician capable of executing the task and currently executing another task with normal priority level is re-assigned to the urgent task, as formalized by rule r16.

3.4 Software Implementation and Technical Details

We implemented the controller as two sub-components: an *execution engine client* (*client* in short) and a *reasoner and fact base manipulator* (*manipulator* in short), as illustrated in Figure 3. This separation improves the modularization and provides flexibility of the implementation – for example, selecting a different reasoner requires only modifications to the manipulator, whereas a change of the execution engine requires only modifications to the client.

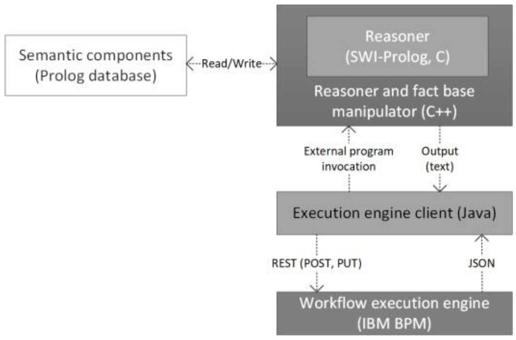

Figure 3. Software implementation of the TWMF

While the TWMF is implementation-agnostic (it can work with any workflow execution engine or any FOL reasoner), we made several choices when developing our proof-of-concept implementation. With regards to the information about clinical workflows and clinicians, the execution engine stores a complete description of clinical workflows using BPMN, while the TWMF stores supplementary information needed for modeling dynamic formation of an IHT, such as capabilities required by workflow tasks or possessed by clinicians.

Earlier we used Microsoft's Z3 solver as the reasoner [28], but in this research we decided to switch to SWI-Prolog [35]. In comparison to Z3, SWI-Prolog allows for running specific queries (e.g., to identify a clinician for assignment to a specific task) and produces output in a format that is easier to parse and analyze. SWI-Prolog is implemented in the C programming language and thus could be easily embedded in the manipulator written in C++. The latter is a stand-alone application that is called by the client with appropriate parameters (e.g., identifiers of tasks) and returns responses generated by SWI-Prolog.

IBM Business Process Manager (BPM) [36] is our execution engine of choice. It is an integrated platform that executes workflows defined in BPMN and provides additional support for mobility and communication (e.g., instant messaging) between clinicians involved in workflow execution.

The client is written in Java and communicates with IBM BPM using its programming interface (RESTful API). Specifically, the client handles creating a workflow, assigning the MRP to the workflow, retrieving a list of completed tasks and next tasks to be executed, and assigning a clinician to a task.

We described the use of API methods for IBM BPM in [37]. We should note here that similar API is also available for other workflow management systems, such as Bonita [38] or Camunda [39], making it possible to replace IBM BPM without extensive revisions to the client (other components of the TWMF do not require modifications).

4 A Proof-of-Concept Use Case

In this section, we illustrate selected operations of the TWMF. We focus on creating an IHT for a new patient admitted to a tertiary care institution, selecting the MRP and team members, and handling urgent task and task collection in a setting where multiple patients are managed simultaneously.

To simplify the description of this use case we skip the workflow associated with the diagnostic phases and assume that execution starts with a condition-specific workflow. We consider here two primary workflows for hospital use: one associated with managing a radical prostatectomy patient (i.e., patient with prostate cancer who is undergoing a surgery) and the other associated with managing a patient having acute stroke. These workflows are presented in Figures 4 and 5, respectively. For better readability, we explicitly indicate workflow invocation instead of embedding child workflows into parent ones. For simplicity, we also assume that each task requires a single capability, although TWMF handles more complex and realistic situations.

We assume that five clinicians (three physicians – WM, SW and DA, and two nurses -- NC and MK) are available candidates for respective IHTs. Clinical specialties, capabilities (relevant for this use case), and expertise levels of these clinicians are presented in Table 2. This information is stored in the fact base. For example, WM's specialty and its level are captured by the fact `hasSpecialty(WM, surgery, 3)`. In Appendix 3, we present additional facts from the fact base and explain how the content of this fact base changes during the IHT operations.

Table 2. Description of clinicians. Expertise levels are encoded using numbers (1 = novice, 2 = regular, 3 = expert)

Clinician	Type	Specialty (expertise level)	Possessed capability (expertise level)
WM	Physician	Surgery (3)	Perform surgery (3), Assist in surgery (3)
SW	Physician	Surgery (1)	Perform surgery (1), Assist in surgery (2)
DA	Physician	Neurology (3)	Assess imaging need (3), Perform MUST test (3)
NC	Nurse	Surgical nursing (2)	Assess drainage (3), Assess GCS (2), Perform MUST test (2)
MK	Nurse	Surgical nursing (2)	Assess drainage (2), Assess GCS (2), Perform instrumentation (2)

4.1 Workflow Selection and Team Creation

When a new patient (named John) with diagnosed prostate cancer is admitted, the controller is notified by the HIS – it updates the fact base and invokes the reasoner, which uses rule r1 to infer that the *radical prostatectomy* workflow should be selected. The controller queries the execution engine to instantiate this workflow for John and assembles a new IHT for managing John's condition.

4.2 MRP Selection

Before the execution of the *radical prostatectomy* workflow can start, the MRP needs to be identified. While rule r6 is not applicable (the team for John has no members), rule r7 identifies two possible candidate physicians – WM and SW. They are both eligible for the *radical prostatectomy* workflow and are available. We assume that due to his earlier experiences John has a positive preference towards SW and is indifferent about other clinicians, so SW becomes the MRP for the current workflow. This selection is recorded in the fact base and the execution engine is notified to proceed with the execution.

At this point no task have been completed and the *admission* task is waiting for clinician assignment. The status of this task is updated in the fact base and the reasoner is invoked. Admission is an MRP-specific task; therefore, according to rule r9, this task is assigned to SW and the execution engine proceeds with its execution. When the *admission* task is executed and its status is updated to *completed*, the *OR (Operating Room) day* child workflow for John is invoked and the parent workflow is suspended.

There are 4 parallel tasks in the *OR day* workflow that need to be executed. However, the MRP for this workflow needs to be identified. Application of rule r6 fails because SW (current MRP) is not eligible for this child workflow (his expertise level is not high enough) and application of rule r7 identifies WM as MRP for the *OR day* workflow.

Once the execution of the *OR day* workflow has been completed, the parent workflow is resumed and SW is automatically restored as MRP. If SW is not available at this time (for example, he is off-shift), then the controller will look for a different MRP.

4.3 Clinician's Assignment

Let us assume that John is in the first day after the surgery and that his team executes two child workflows (*day 1* and *assessment*). The *drainage assessment* task is on the list of to-be-executed tasks reported by the execution engine. There are two nurses who have the required capability with the appropriate expertise level: NC and MK (see Table 4). Let's assume that MK was recruited as a team member before (she was responsible for *instrumentation* during the surgery), but now she is busy with an urgent task from another workflow. Rule r10 cannot be triggered and rule r11 suggests NC as a possible candidate for assignment to drainage assessment task. This is recorded in the fact base and the execution engine is notified to proceed with the execution.

At this point, another patient named Mary, who is having an acute stroke, has been admitted. A different IHT has been created, the acute stroke workflow has been initiated, and DA has been selected as MRP. Now, there is a need to assign a nurse or a physician to the *GCS* (*Glasgow Coma Scale*) *assessment* task. The task is marked as urgent and is part of a task collection. The best candidate for executing the task would be NC (she is capable of executing all tasks from this collection); however, she is busy executing the normal priority *drainage assessment* task in the *radical prostatectomy* workflow and rule r13 fails to produce the assignment. MK is another candidate capable of executing this task (although she is not capable to execute other tasks from this collection). She is however busy with other urgent task and rule r15 fails to produce the assignment. Rule r16 is now applied and it requests to reassign NC from *drainage assessment* for John to *GCS assessment* for Mary (the status of *drainage assessment* is changed to *clinician assignment*, while status of *GCS assessment* is revised to *being executed)*.

The execution of the *acute stroke* workflow for Mary continues until the *MUST (Malnutrition Universal Screening Tool) test* task is encountered. This task belongs to the same collection as *GCS assessment*, thus NC would be the most appropriate candidate to be assigned. However, let's assume NC has gone off-shift and another clinician needs to be found. As rule r12 fails to produce the assignment, application of rule r13 identifies DA as the candidate for this task. He has not executed any tasks from this collection, but is the MRP for the *acute stroke* workflow for Mary and is assigned to execute *MUST test*.

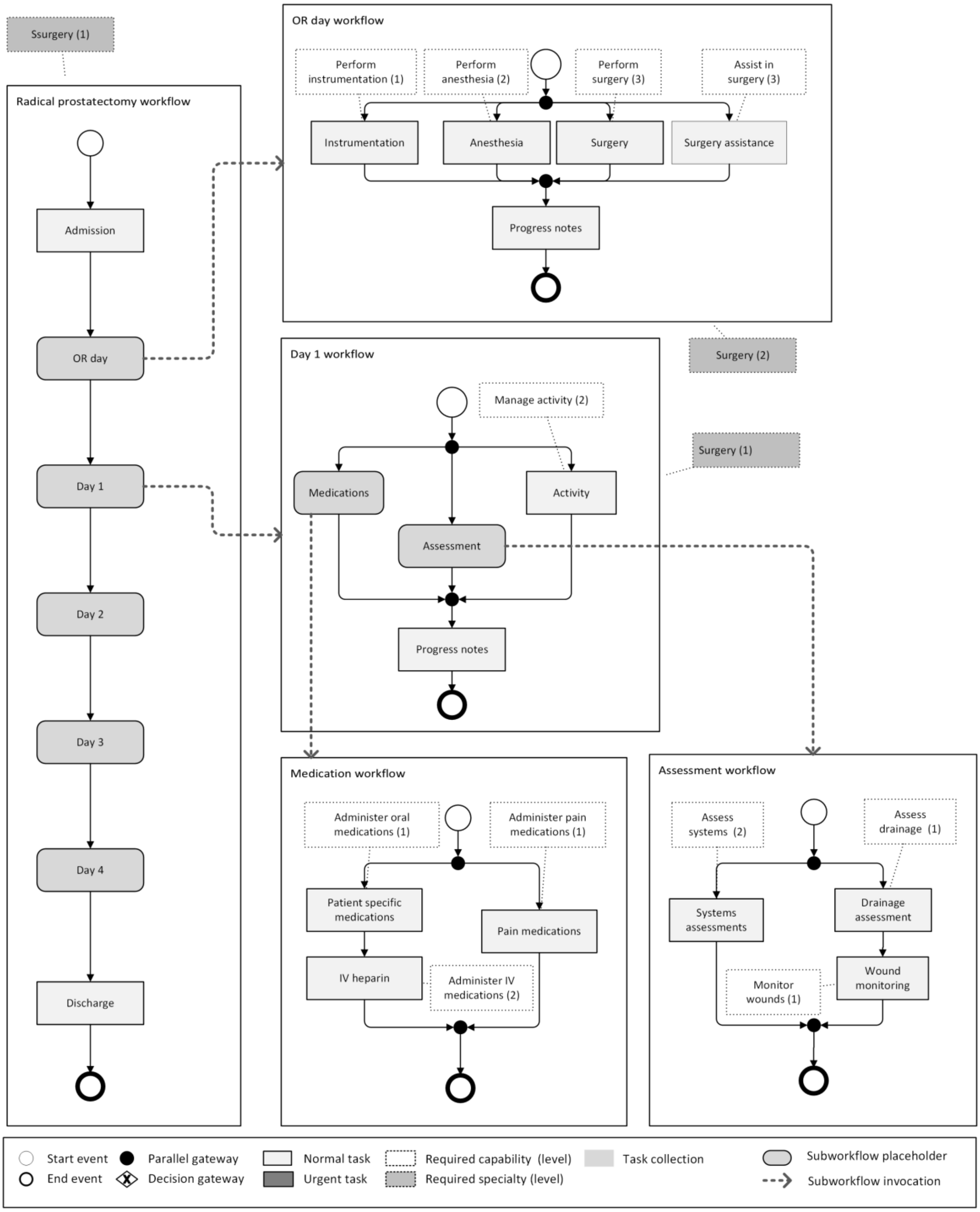

Figure 4. Example radical prostatectomy workflow

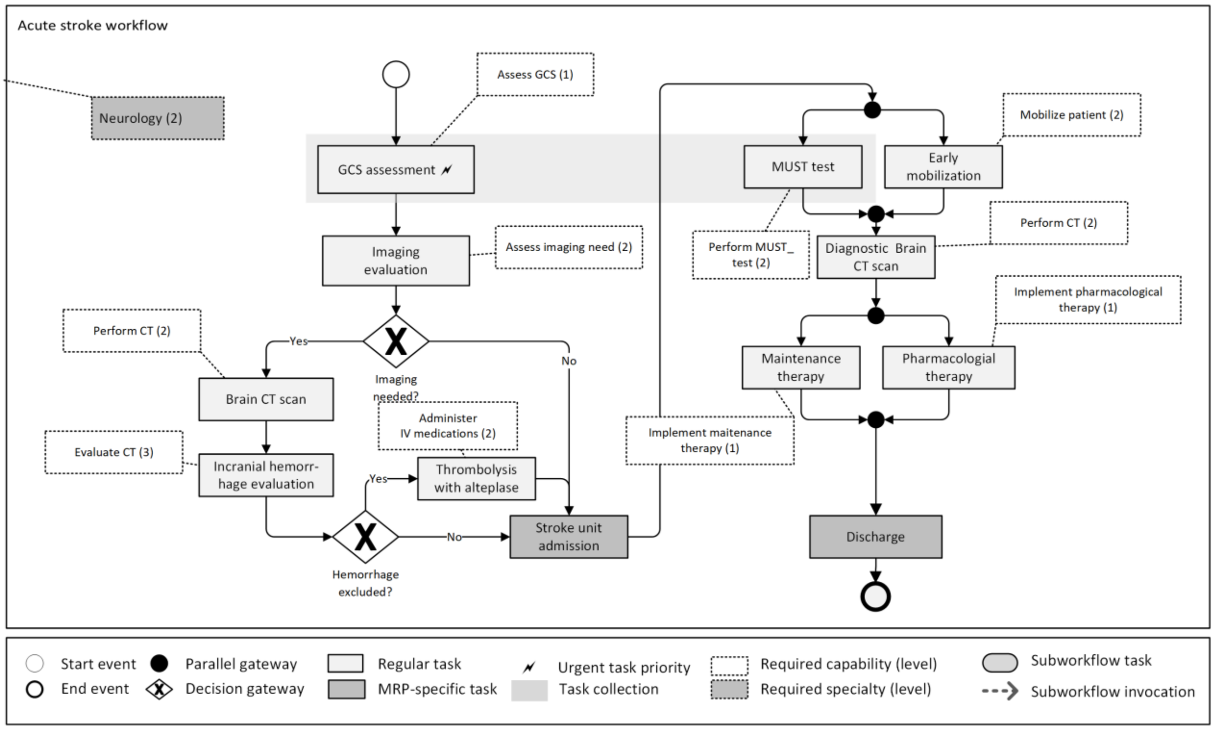

Figure 5. Example acute stroke workflow

5 Discussion

In this paper, we presented the conceptual foundations, design, and proof-of-concept implementation of the TWMF that significantly extends our earlier research on the MET4 system [15, 27]. While MET4 was a self-contained and stand-alone system that could not integrate with different workflow execution technologies, the TWMF is designed to operate independently of these technologies, taking full advantage of them, and extending them to accommodate the demands of modeling and executing healthcare processes. Our proposed approach is different from [40] where solutions aiming to provide flexibility to workflow execution were developed in isolation from existing business workflow models.

We envision the use of the TWMF in integrated or simulated modes. In the integrated mode, illustrated in this paper, the framework interacts with external systems in order to support real-time execution of clinical workflows. In the simulated mode, the TWMF is used to assess the impact of different conditions (staffing patterns, different team dynamics, etc.) on IHT dynamic formation and the framework is used to simulate possible executions of workflows.

We assessed the TWMF by creating a proof-of-concept implementation and applying it to hypothetical use case involving two clinical workflows. This use case was developed by clinical experts and executed in a laboratory setting for comprehensive evaluations. The experts were presented with detailed logs describing IHT operations and attested that the modeled dynamic formation of a team is correct from a practical perspective.

The methodological solutions proposed in this paper come with limitations:

1. TWMF covers selected aspects of dynamic formation of an IHT, and the proof-of-concept implementation involved execution of a limited number of workflows. While in test runs in a laboratory setting with simulated multiple workflows and simulated multiple patients, we did not encounter any processing delays, this might not be true in a live environment. Moreover, we did not consider the impact of such solution on the efficient use of personnel and delays in task execution. Our future work will continue along the system-testing continuum developed by Kushniruk et al. [29] and involve evaluating the TWMF with interdisciplinary healthcare teams in realistic clinical settings.
2. Although there are no theoretical reasons why TWMF could not be ported to notations other than BPMN (for example, those used for computer-interpretable guidelines like PROforma [41]), this assertion should be tested.
3. TWMF assumes one-to-one mapping between a condition (or presentation) and a team. In case of a complex patient with multimorbidity this will result in creating multiple teams, although some clinicians may participate in many teams, thus limiting the perceived redundancy in representation. This limitation could be addressed by adopting techniques for merging computer-interpreted guidelines (CIGs) [42] and for mitigating adverse interactions between CIGs [43]. These techniques could be used for safely (i.e., avoiding conflicts in patient management) combining multiple clinical workflows required for managing a complex patient. Once such a combined workflow has been established, it could be processed by TWMF.

4. In our proposed framework, clinical decision making is handled at the workflow level by the execution engine, thus support for complex decision problems that require shared decisions is dependent on the underlying IT infrastructure and its capabilities. Earlier we proposed ontology-based formalization of shared decision making process [44] that can be integrated into the TWMF. Such integration asks for extending the IHT ontology and transfering some functionalities from the execution engine to the controller. Thus, after encountering a shared decision step, the execution engine would send a request to the controller. In response, the controller would use some shared decision making model to establish a decision (possibly interacting with the patient and relevant IHT members) and report this decision to the execution engine. Based on the decision outcome, the execution engine would select one of possible execution paths in the workflow.
5. TWMF relies on the execution engine to support workload management across the teams. While such solution allows for capturing up-to-date status of clinicians, it is insufficient to optimize and balance the workload (e.g., it is possible that a clinician preferred by several patients may have a higher workload than others). To address this limitation, it is possible to improve the process of selecting the MRP or a team member by formulating it as a two-criteria optimization problem with one criterion operating on patient preferences and the other one operating on cumulative workload of specific physician. Implementing this solution would require the execution engine to keep track of the workload and to report it when requested to the controller.

We plan to address these limitations as part of our future work. We are also planning to extend the TWMF applicability into primary care outpatient management. This will require changing the conceptual foundations and introducing corresponding revisions to the IHT ontology and the formation rules. However, we believe we will be able to re-use the framework architecture without any significant structural changes. Along this line of research, we are planning to consider information about previous visits of the same patient and formalize coordination and planning strategies to manage the patient transition between a tertiary care team and a community-based primary care IHT.

We also plan to incorporate patient preferences with regards to the treatment options – a possible solution we presented in [27]. Incorporating this solution will require revising the IHT ontology and transferring selected functionalities from the execution engine to the controller, similarly solution discussed in point 4 above. Furthermore, we plan to expand the TWMF by considering family members and their preferences, as well as data access privileges and requirements. The former should provide better support for participatory care [45], while the latter should result in a hybrid control system that checks whether a clinician is not only capable to execute a task, but also whether she/he has sufficient rights to access information that is required to complete this task. Finally, we will be also looking at developing methods for handling patient- and workflow-related exceptions possibly using libraries of exception handlers [46].

Summary Table

What was already known on this topic:

- Management of complex conditions by an interdisciplinary healthcare team (IHT) results in improved health outcomes.
- IHT operationalization requires a health information technology system redesign to accommodate dynamic formation of a team.
- Research on IHT dynamic formation has focused on *how* to model certain aspects of team dynamics instead of *what* aspects to model.

What this study added to our knowledge:

- Dynamic formation of an IHT can be modeled and operationalized with ontology-driven design and logic-based reasoning realized as the Team and Workflow Management Framework (TWMF).
- Business workflow models may be expanded with the knowledge of team formation that is specific for healthcare setting.
- TWMF can be implemented as a software layer on the top of a business workflow modeling and execution engine and integrated with a hospital information system (HIS).

Acknowledgements

The authors would like to thank the reviewers for their insightful comments that helped to improve the paper.

This research was supported by grants from the NSERC CREATE Program in Healthcare Operations and Information Management, NSERC Discovery Grant Program, and the Telfer School of Management Research Fund. The research was conducted when Dr. Kezadri-Hamiaz and Dr. Rosu were postdoctoral fellows with the MET Research Group at the Telfer School of Management, University of Ottawa. We would like to thank physicians from The Ottawa Hospital for their help in developing and validating different patterns of a dynamic formation of an IHT, explaining clinical workflows, and assisting with the evaluation of proof-of-concept prototypes.

References

1. Hall P, Weaver L. Interdisciplinary education and teamwork: a long and winding road. Med Educ 2001; 35 (9): 867-75.
2. Stokols D, Hall KL, Taylor BK, Moser RP. The science of team science: overview of the field and introduction to the supplement. Am J Prev Med 2008; 35 (2 Suppl): S77-89.
3. Casimiro LM, Hall P, Kuziemsky C, O'Connor M, Varpio L. Enhancing patient-engaged teamwork in healthcare: an observational case study. J Interprof Care 2015; 29 (1): 55-61.
4. Nancarrow SA, Booth A, Ariss S, Smith T, Enderby P, Roots A. Ten principles of good interdisciplinary team work. Hum Resour Health 2013; 11: 19.
5. Gordon RM, Corcoran JR, Bartley-Daniele P, Sklenar D, Sutton PR, Cartwright F. A transdisciplinary team approach to pain management in inpatient health care settings. Pain Manag Nurs 2014; 15 (1): 426-35.
6. Kuziemsky C. Decision-making in healthcare as a complex adaptive system. Healthc Manage Forum 2016; 29 (1): 4-7.
7. Rad AA, Benyoucef M, Kuziemsky C. An Evaluation Framework for Business Process Modeling Languages in Healthcare. Journal of Theoretical and Applied Electronic Commerce Research 2009; 4 (2): 1-19.
8. Karsh BT, Weinger MB, Abbott PA, Wears RL. Health information technology: fallacies and sober realities. J Am Med Inform Assoc 2010; 17 (6): 617-23.
9. Grace SM, Rich J, Chin W, Rodriguez HP. Implementing Interdisciplinary Teams Does Not Necessarily Improve Primary Care Practice Climate. Am J Med Qual 2016; 31 (1): 5-11.
10. Mabry SL, Schneringer T, Etters T, Edwards N. Intelligent Agents for Patient Monitoring and Diagnostics. In; 2003 2003: ACM; 2003. p. 257–262.
11. Taboada M, Cabrera E, Iglesias ML, Epelde F, Luque E. An Agent-Based Decision Support System for Hospitals Emergency Departments. In; 2011 January 1, 2011; 2011. p. 1870-1879.
12. Grosz BJ. A Multi-agent Systems "Turing Challenge". In; 2013 2013: International Foundation for Autonomous Agents and Multiagent Systems; 2013. p. 3–4.
13. Huguet L, Sabouret N, Lourdeaux D. 'Errare Humanum Est': Simulation of communication error among a virtual team in crisis situation. In; 2017 2017; 2017. p. 344-350.
14. Ruan J, MacCaull W, Jewers H. Enhancing Patient-Centered Palliative Care with Collaborative Agents. In; 2010 2010: IEEE Computer Society; 2010. p. 356–360.

15. Wilk S, Kezadri-Hamiaz M, Rosu D, Kuziemsky C, Michalowski W, Amyot D, et al. Using Semantic Components to Represent Dynamics of an Interdisciplinary Healthcare Team in a Multi-Agent Decision Support System. J Med Syst 2016; 40 (2): 42.
16. van der Aalst WP, ter Hofstede AM, Weske M. Business Process Management: A Survey. In: van der Aalst WP, Weske M, editors. Business Process Management: Springer Berlin Heidelberg; 2003. p. 1-12.
17. Ozkaynak M, Brennan PF, Hanauer DA, Johnson S, Aarts J, Zheng K, et al. Patient-centered care requires a patient-oriented workflow model. J Am Med Inform Assoc 2013; 20 (e1): e14-6.
18. Lanzén A, Oinn T. The taverna interaction service: enabling manual interaction in workflows. Bioinformatics 2008; 24 (8): 1118-1120.
19. Ludäscher B, Altintas I, Berkley C, Higgins D, Jaeger E, Jones M, et al. Scientific workflow management and the Kepler system. Concurrency and Computation: Practice and Experience 2006; 18 (10): 1039-1065.
20. Grando MA, Peleg M, Cuggia M, Glasspool D. Patterns for collaborative work in health care teams. Artif Intell Med 2011; 53 (3): 139-60.
21. Papapanagiotou P, Fleuriot JD. Formal verification of collaboration patterns in healthcare. Behaviour and Information Technology 2014; 33 (12): 1278-1293.
22. Sims S, Hewitt G, Harris R. Evidence of collaboration, pooling of resources, learning and role blurring in interprofessional healthcare teams: a realist synthesis. J Interprof Care 2015; 29 (1): 20-5.
23. Andreatta PB. A typology for health care teams. Health Care Management Review 2010; 35 (4): 345-354.
24. Sims S, Hewitt G, Harris R. Evidence of a shared purpose, critical reflection, innovation and leadership in interprofessional healthcare teams: a realist synthesis. J Interprof Care 2015; 29 (3): 209-15.
25. West MA, Lyubovnikova J. Illusions of team working in health care. J Health Organ Manag 2013; 27 (1): 134-42.
26. Xiao Y, Henrickson Parker S, Manser T. Teamwork and collaboration. Reviews of Human Factors and Ergonomics 2013; 8 (1): 55-102.
27. Kuziemsky C, Astaraky D, Wilk S, Michalowski W, Andreev P. A framework for incorporating patient preferences to deliver participatory medicine via interdisciplinary healthcare teams. AMIA Annual Symposium Proceedings 2014: 835-844.
28. Wilk S, O'Sullivan D, Kezadri-Hamiaz M, Kuziemsky C, Rosu D, Michalowski W, et al. Aligning interdisciplinary healthcare team behavior with workflow execution: An example of a radical prostatectomy workflow. In: 2016 IEEE 29th International Symposium on Computer-Based Medical Systems (CBMS): IEEE; 2016. p. 112-117.
29. Kushniruk A, Nohr C, Jensen S, Borycki EM. From Usability Testing to Clinical Simulations: Bringing Context into the Design and Evaluation of Usable and Safe Health Information Technologies. Contribution of the IMIA Human Factors Engineering for Healthcare Informatics Working Group. Yearb Med Inform 2013; 8: 78-85.
30. Kuhn DR, Coyne EJ, Weil TR. Adding Attributes to Role-Based Access Control. Computer 2010; 43 (6): 79-81.
31. Křemen P, Kouba Z. Ontology-driven information system design. IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews 2012; 42 (3): 334-344.
32. Sahoo SS, Ramesh P, Welter E, Bukach A, Valdez J, Tatsuoka C, et al. Insight: An ontology-based integrated database and analysis platform for epilepsy self-management research. Int J Med Inform 2016; 94: 21-30.
33. Smullyan RM. First-order logic: Springer; 1968.
34. Gruber TR. Toward principles for the design of ontologies used for knowledge sharing? International Journal of Human-Computer Studies 1995; 43 (5): 907-928.
35. Wielemaker J, Schrijvers T, Triska M, Lager T. SWI-Prolog. Theory and Practice of Logic Programming 2012; 12 (1-2): 67-96.
36. IBM Business Process Manager. Cited February 10, 2019. Available from: https://www.ibm.com/automation/software/workflow.

37. Çatal N, Amyot D, Michalowski W, Kezadri-Hamiaz M, Baslyman M, Wilk S, et al. Supporting process execution by interdisciplinary healthcare teams: Middleware design for IBM BPM. Procedia Computer Science 2017; 113 (Supplement C): 376-383.
38. Bonita BPM. Cited February 10, 2019. Available from: http://www.bonitasoft.com/.
39. Camunda. Cited February 10, 2019. Available from: https://camunda.org/.
40. Lam J, Guerin F, Vasconcelos W, Norman TJ. Building multi-agent systems for workflow enactment and exception handling. In: Coordination, Organizations, Institutions and Norms in Agent Systems V: Springer; 2010. p. 53-69.
41. Peleg M. Computer-interpretable clinical guidelines: a methodological review. J Biomed Inform 2013; 46 (4): 744-63.
42. Riaño D, Collado A. Model-Based Combination of Treatments for the Management of Chronic Comorbid Patients. In: Peek N, Marín Morales R, Peleg M, editors. Artificial Intelligence in Medicine. 14th Conference on Artificial Intelligence in Medicine, AIME 2013, Murcia, Spain, May 29 – June 1, 2013. Proceedings: Springer; 2013. p. 11-16.
43. Wilk S, Michalowski M, Michalowski W, Rosu D, Carrier M, Kezadri-Hamiaz M. Comprehensive mitigation framework for concurrent application of multiple clinical practice guidelines. J Biomed Inform 2017; 66: 52-71.
44. Parimbelli E, Wilk S, Kingwell S, Andreev P, Michalowski W. Shared Decision-Making Ontology for a Healthcare Team Executing a Workflow, an Instantiation for Metastatic Spinal Cord Compression Management. AMIA Annu Symp Proc 2018; 2018: 877-886.
45. Whiston L, Barry JM, Darker CD. Participation of patients and family members in healthcare services at the service and national levels: A lesson learned in Dublin, Ireland. Patient Educ Couns 2017; 100 (3): 583-591.
46. Grando A, Peleg M, Glasspool D. A goal-oriented framework for specifying clinical guidelines and handling medical errors. Journal of Biomedical Informatics 2010; 43 (2): 287-299.

Appendix 1

The ontology (presented in Figure 2) is partitioned into three sub-ontologies, each containing concepts and relations pertinent to workflow, patient, and a team. These sub-ontologies are related through the shared concepts of: *Condition*, *Capability*, *Specialty* and *ExpertiseLevel*. The *Condition* concept captures the presentation or condition the patient is being managed for, the *Capability* concept captures the ability of a clinician to perform a clinical task, while the *Specialty* concept captures her/his clinical specialty. Finally, *ExpertiseLevel* concept captures the level of expertise (we assume three levels – novice, normal or expert) associated with the concepts of *Capability* and *Specialty*.

A1.1 Workflow Sub-ontology

This sub-ontology introduces concepts and relations associated with a workflow. Its purpose is to support seamless exchange of workflow-related information between the controller and the execution engine and to expand the workflow representation used by the execution engine with new concepts required to model dynamic formation of IHT. The central concepts in this sub-ontology are *Workflow* and *Task*.

A workflow is associated with a patient condition through the *managesCondition* relation – this relation is optional and it does not have to be specified for condition-independent workflows, such as for example the ABCDE (airway, breathing, circulation, disability, exposure) workflow typically invoked for trauma patients. The association between a workflow, clinician’s specialty and expertise level required for the MPR role is captured with the *requiresSpecialty* relation. This relation does not need to be specified for a child workflow (see description of formation rule r8 for details).

A workflow is associated with IHT through the *executesWorkflow* relation and with workflow execution status, represented by the *WorkflowStatus* concept, through the *hasWorkflowStatus* relation. Possible workflow status values are: *MRP selection* (a workflow is waiting for MRP selection – this is a default value), *being executed* and *completed*. Parent and child workflows are associated through the *invokesWorkflow* relation established from the output generated by the execution engine.

A task is characterized by capabilities and associated minimal expertise levels and this is modeled by the *requiresCapability* relation. A task is also characterized by a level of priority (*normal* or *urgent*) represented by the concept *PriorityLevel* and linked to the task through the *hasPriority* relation. A task has a status captured by the *TaskStatus* concept and linked by the *hasTaskStatus* relation. Possible task status values include *waiting for execution* (default value), *clinician assignment* (a task is waiting to be assigned to a clinician), *being executed* and *completed*. Finally, a task is associated with a workflow through the *isPartOfWorkflow* relation.

If there are multiple tasks that should be executed by the same clinician, they are organized into a collection represented by the *TaskCollection* concept. The association between a task and a collection is modeled by the *isPartOfTaskCollection* relation. MRP-specific tasks do not need to have capabilities explicitly stated and they are not included in any other task collection – in fact they form an implicit collection of all tasks to be assigned to the MRP

A1.2 Patient Sub-ontology

This sub-ontology introduces concepts and relations associated with a patient. In comparison to our earlier work [15], the patient sub-ontology has been simplified as the responsibilities associated with clinical data management and decision making are transferred to the execution engine. A patient, represented by the *Patient* concept, is managed for a given condition (modeled by the *hasCondition* relation) and is associated with a primary workflow (*hasPrimaryWorkflow* relation). The latter is an example of an inferred relation that is established automatically through the inference of formation rules.

In order to model patient preferences, we introduced the *hasPreference* relation that captures preference a patient has for a specific clinician who treats her/him. The relation associates a patient and a clinician with a preference level represented by the *PreferenceLevel* concept. Currently we consider three possible preference levels: *negative*, *neutral* and *positive*, with *neutral* being the default one. *Positive* preference level indicates that a patient would prefer to be managed by a clinician, while *negative* level indicates that a patient would like to avoid a clinician.

This *hasPreference* relation is considered when selecting an MRP or assigning a clinician to a task. It can be defined explicitly (e.g., from responses provided by a patient when the care management starts) or implicitly by applying a more complex preference model (e.g., a value function).

A1.3 Team Sub-ontology

This sub-ontology contains concepts and relations associated with an IHT structure represented by the *Team* concept. A team consists of multiple clinicians (physicians, nurses, allied health professionals, collectively modeled by the *Clinician* concept) working together to manage a patient. The *managesPatient* relation captures the connection between an IHT and a patient. A team executes one or more workflows and this is captured by the *executesWorkflow* relation.

Each clinician is characterized by capabilities (one or more) and specialties with their associated expertise levels. This is captured by the *hasCapability* and *hasSpecialty* relations, respectively. The clinician’s status is represented by the *ClinicianStatus* concept. Possible status values are *free*, *busy* or *off-shift*. The association between a clinician and her/his status is modeled by the *hasClinicianStatus* relation.

Information about team membership is captured by the *isMRP* and *hasAssignment* relations. The former indicates that the clinician acts as the MRP for a workflow. Since the workflow is associated with the team that executes it (through *executesWorkflow*), there is also an indirect relation between the MRP and the team. The *hasAssignment* relation associates the clinician with the task she/he is assigned to execute. The task is associated with the workflow (through *isPartOfWorkflow*).

The *shouldChangeAssignment* relation associated with two tasks signals that the assignment of these tasks to clinicians needs to be changed. This occurs when a clinician who is busy with the execution of a task having a normal priority level is requested to execute an unassigned task with an urgent priority level. Specifically, this relation implies the status of the first task (normal priority level) should be changed to *clinician assignment*, and the status of second task (urgent priority level) should be changed to *being executed*.

Appendix 2

Here we write the formation rules described in Section 3.3 using a FOL-based syntax. Each rule has a form of *antecedent → consequent*, where the antecedent is a conjunction of conditions that need to be satisfied so the consequent holds. The antecedent is evaluated using the content of the fact base and the consequent is added to the fact base. Both antecedent and consequent are expressed using concepts and relations from the IHT ontology. In order to simplify the notation instead of universal qualifiers in rule antecedents we use the ***forall*** construct inspired by Prolog and we skip existential quantifiers.

In the rules we use the following symbols to indicate specific concepts in the IHT ontology: *c = Capability*, *cl = Clinician*, *cls = ClinicianStatus*, *co = Condition*, *el = ExpertiseLevel*, *pt = Patient*, *s = Specialty*, *t = Task*, *tc = TaskCollection*, *tm = Team* and *w = Workflow.*

	Rule
r1	$hasCondition(pt, co) \wedge managesCondition(w, co) \rightarrow \ hasPrimaryWorklow(pt, w)$
r2	$isMRP(cl, w) \wedge executesWorkflow(tm, w) \rightarrow isPastOrCurrentMember(cl, tm)$
r3	$isAssigned(cl, t) \wedge isPartOfWorkflow(t, w) \wedge executesWorkflow(tm, w) \rightarrow isPastOrCurrentMember(cl, tm)$
r4	$\boldsymbol{forall}\big(requiresCapability(t, c, el_t) \rightarrow hasCapability(cl, c, el_{cl}) \wedge (el_{cl} \geq el_t)\big) \rightarrow isCapableOfTask(cl, t)$
r5	$hasSpeciality(cl, s, el_{cl}) \wedge requiresSpecialty(w, s, el_w) \wedge el_{pr} \geq el_w \rightarrow isEligibleForWorkflow(cl, w)$
r6	$executesWorkflow(tm, w) \wedge isEligibleForWorkflow(cl, w) \wedge hasClinicianStatus(cl, cls) \wedge (cls \neq OFF\_SHIFT) \wedge isPastOrCurrentMember(cl, tm) \rightarrow isCandidateForMRP(cl, w)$
r8	$\boldsymbol{not}\big(requiresSpecialty(w, s, el)\big) \wedge invokesWorkflow\big(w_{parent}, w\big) \wedge isMRP\big(cl, w_{parent}\big) \rightarrow isCandidateForMRP(cl, w)$
r9	$isPartOfWorkflow(t, w) \wedge \boldsymbol{not}\big(requiresCapability(t, c, el)\big) \wedge isMRP(cl, w) \rightarrow isCandidateForAssignment(cl, t)$
r10	$\boldsymbol{not}\big(isPartOfCollection(t, tc)\big) \wedge requiresCapability(t, c, el) \wedge isCapableOfTask(cl, t) \wedge isPartOfWorkflow(t, w) \wedge executesWorkflow(tm, w) \wedge isPastOrCurrentMember(tm, cl) \wedge hasClinicianStatus(cl, FREE) \rightarrow isCandidateForAssignment(cl, t)$
r12	$isPartOfCollection(t, tc) \wedge isCapableOfTask(cl, t) \wedge hasClinicianStatus(cl, FREE) \wedge isPartOfCollection\big(t_{prev}, tc\big) \wedge \big(t_{prev} \neq t\big) \wedge hasAssignment\big(cl, t_{prev}\big) \wedge \boldsymbol{forall}\big(isPartOfCollection(t_{next}, tc) \wedge hasTaskStatus(t_{next}, WAITING\_FOR\_EXECUTION) \rightarrow isCapableOfTask(cl, t_{next})\big) \rightarrow isCandidateForAssignment(cl, t)$
r16	$hasTaskPriority\big(t_{urgent}, URGENT\big) \wedge \boldsymbol{not}\Big(isEligibleForTask\big(cl, t_{urgent}\big) \rightarrow hasClinicianStatus(cl, FREE)\Big) \wedge isCapableOfTask\big(cl, t_{urgent}\big) \wedge hasAssignment(cl, t_{normal}) \wedge hasPriority(t_{normal}, NORMAL) \wedge hasTaskStatus(t_{normal}, BEING\_EXECUTED) \rightarrow shouldChangeAssignment\big(t_{normal}, t_{urgent}\big)$

Appendix 3

Here we provide more detailed description into the interactions among the TWMF components, and we illustrate the fact base and how its content changes during workflows' execution. This complements earlier presentation of the use case by focusing on interactions between the controller and the reasoner and on changes introduced to the fact base.

When presenting sample facts in the table below, we use labels of clinicians from Table 2. Names of workflows and tasks are consistent with the ones introduced in Figure 4 and Figure 5 – when required we add suffixes with patient names to emphasize processing patient-specific copies of workflows and tasks. For the sake of brevity, we do not present all facts that are introduced when reasoning with formation rules, but focus on the most relevant ones. We do not present most of the facts that are processed by the controller in order to establish final recommendation for the execution engine.

Workflow selection and team creation

1. The controller stores information about John's condition (fact f1) and invokes the reasoner to identify the primary workflow for John. The reasoner applies rule r1 to facts f1 and f2 and selects the *radical prostatectomy* workflow (fact f3).
2. The controller creates a new team to manage John and associates this team with John-specific copy of the workflow (facts f4 and f5).

MRP selection

1. The primary workflow for John requires the MRP (fact f6). The reasoner uses rule r7 to identify possible candidates for MRP (facts f7 and f8), Based on John's preferences (fact f9), the reasoner selects SW as MRP for the workflow (fact f10).
2. The controller updates the status of the *admission* task for John (fact f11) and invokes the reasoner to establish the assignment. Based on rule r9 this task is assigned to SW (fact f12).
3. Once the *admission* task has been completed by the execution engine, the controller changes its status (fact f13).

4. The controller stores information that the child workflow *OR day* has been invoked by the primary workflow and that it asks for the MRP assignment (facts f14 and f15), and then invokes the reasoner. The reasoner uses rule r7 to select WM as MRP candidate for the child workflow and selection is accepted by the controller (fact 16).

Clinician's assignment

1. The controller changes the status of the *drainage assessment* task for John (fact f17) and invokes the reasoner. Rule r11 suggests NC as a candidate for assignment to this task and this is confirmed by the controller (fact f18).
2. On Mary's admission the controller stores information about her condition (f19), assembles new team to manage Mary (fact f20) and associates this team with a Mary-specific copy of the *acute stroke* workflow (fact f21). DA is selected as MRP for this workflow (fact f22).
3. The controller changes the status of the *GCS assessment* task for Mary (fact f23) – this task is urgent (fact f24) and belongs to a task collection (fact f25). The reasoner is invoked and rule r16 suggests reassignment of this task and *drainage assessment* (fact f26). The assignment process is repeated and the task is now assigned to NC (fact f27).
4. The controller changes the status of the *MUST test* task for Mary (fact f28). This task belongs to the same collection as *GCS assessment* (fact f29). The reasoner is invoked -- since NC is off-shift (fact f30), rule r13 selects DA as a candidate for assignment. This selection is accepted by the controller (fact f31).

	Fact in the fact base
f1	hasCondition(John, prostate-cancer)
f2	managesCondition(radical-prostatectomy, prostate-cancer)
f3	hasPrimaryWorkflow(John, radical-prostatectomy)
f4	managesPatient(team-1, John)
f5	executesWorkflow (team-1, radical-prostatectomy-John)
f6	hasWorkflowStatus(radical-prostatectomy-John, MRP_SELECTION)
f7	isCandidateForMRP(WM, radical-prostatectomy-John)
f8	isCandidateForMRP(SW, radical-prostatectomy-John)
f9	hasPreference(John, SW, POSITIVE)
f10	isMRP(SW, radical-prostatectomy-John)
f11	hasTaskStatus(admission-John, CLINICIAN_ASSIGNMENT)
f12	hasAssignment(SW, admission-John)
f13	hasTaskStatus(admission-John, COMPLETED)
f14	invokesWorkflow (radical-prostatectomy-John, OR-day-John)
f15	hasWorkflowStatus(OR-day-John, MRP_SELECTION)
f16	isMRP(WM, OR-day-John)
f17	hasTaskStatus(drainage-assessment-John, CLINICIAN_ASSIGNMENT)
f18	hasAssignment(NC, drainage-assessment-John)
f19	hasCondition(Mary, acute-stroke)
f20	managesPatient(team-2, Mary)
f21	executesWorkflow (team-2, acute-stroke-Mary)
f22	isMRP(DA, acute-stroke-Mary)
f23	hasTaskStatus(GCS-assessment-Mary, CLINICIAN_ASSIGNMENT)
f24	hasPriority(GCS-assessment-Mary, URGENT)
f25	isPartOfCollection(GCS-assessment-Mary, collection-1)
f26	shouldChangeAssignment(drainage-assessment-John, GCS-assessment-Mary)
f27	hasAssignment(NC, GCS-assessment-Mary)
f28	hasTaskStatus(MUST-test-Mary, CLINICIAN_ASSIGNMENT)
f29	isPartOfCollection(MUST-test-Mary, collection-1)
f30	hasClinicianStatus(NC, OFF_SHIFT)
f31	hasAssignment(DA, MUST-test-Mary)