# DCCSS
## *A Meta-model for Dynamic Clinical Checklist Support Systems*

Shan Nan[1,4], Pieter Van Gorp[1], Hendrikus H. M. Korsten[2,1], Uzay Kaymak[1], Richard Vdovjak[3], Xudong Lu[4] and Huilong Duan[4]

[1]*School of Industrial Engineering and Innovation Science, Eindhoven University of Technology, Eindhoven, The Netherlands*
[2]*Department of Aneasthiaology and Intensive-Care, Catharina Ziekenhuis Eindhoven, Eindhoven, The Netherlands*
[3]*Philips Research, Eindhoven, The Netherlands*
[4]*School of Biomedical Engineering and Instrumental Science, Zhejiang University, Zhejiang, China*

Keywords: Checklist, Meta-model, Process, Rule, BPMN, GLIF.

Abstract: Clinical safety checklists receive much research attention since they can reduce medical errors and improve patient safety. Computerized checklist support systems are also being developed actively. Such systems should individualize checklists based on information from the patient's medical record while also considering the context of the clinical workflows. Unfortunately, the form definitions, database queries and workflow definitions related to dynamic checklists are too often hard-coded in the source code of the support systems. This increases the cognitive effort for the clinical stakeholders in the design process, it complicates the sharing of dynamic checklist definitions as well as the interoperability with other information systems. In this paper, we address these issues by contributing the DCCSS meta-model which enables the model-based development of dynamic checklist support systems. DCCSS was designed as an incremental extension of standard meta-models, which enables the reuse of generic model editors in a novel setting. In particular, DCCSS integrates the Business Process Model and Notation (BPMN) and the Guideline Interchange Format (GLIF), which represent best of breed languages for clinical workflow modeling and clinical rule modeling respectively. We also demonstrate one of the use cases where DCCSS has already been applied in a clinical setting.

# 1 INTRODUCTION

Safety checklists have been published by clinical societies in recent years with the purpose of reducing medical errors and improving patient safety. Well established clinical studies indicate these checklists can improve the quality of care significantly (Babayan, 2011). To motivate the routine use of checklists, various checklist support systems have been developed (Avrunin et al., 2012). Among these systems, computerized systems are considered as the most promising ones as (1) they can be integrated with clinical information systems and embedded in the clinical workflow and (2) they can take into account patient-specific context information. Yet, a major obstacle of implementing checklist in computerized support systems is the additional efforts on encoding checklist into the systems, especially making them adaptive to clinical workflow and patient-specific context information.

Currently, computerized checklists are typically encoded directly in the source code of software systems. This is undesirable, especially since such an encoding can be realized in multiple ways. This causes two problems. First of all, ad-hoc encodings in source code increase the time to understand and update the checklist definitions. Secondly, it is unclear how the checklist definitions relate to other clinical software components. For example, it is unclear how hard coded checklists should be integrated with the clinical workflow definitions which are supported by a hospital information system, or with the clinical rules which are supported by clinical decision support systems. Another limitation is that hard coded computerized checklists are only developed for a specific purpose and are difficult to be reused in one hospital or shared among hospitals. This maps to our primary research question: "Which meta-model can sup-


Nan S., Van Gorp P., H. M. Korsten H., Kaymak U., Vdovjak R., Lu X. and Duan H..
DCCSS - A Meta-model for Dynamic Clinical Checklist Support Systems.
DOI: 10.5220/0005241902720279
In *Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development* (MODELSWARD-2015), pages 272-279
ISBN: 978-989-758-083-3

port the modeling of computerized checklists, which are aligned with clinical workflows and which take into account patient-specific context?" While this research question was not tackled by other studies yet, (Färber et al., 2007) have already contributed partial results which we have reused in our study. While (Färber et al., 2007) have already demonstrated the meaningful use of parallel tasks for the modeling of checklist-supported workflows, it was not yet investigated how to deal for example with dynamicity in checklist forms. Also, unlike the results from this paper, previous studies did not yet contribute a meta-model which supports interoperability with workflow management and decision support systems.

Checklists are usually adopted from clinical guidelines and clinical pathways (Weiser et al., 2010). Representation formats of executable clinical guidelines and clinical pathways have been studied for decades (De Clercq et al., 2004). Efforts have been made on formalizing, executing and standardizing clinical processes and clinical rules. Industry standards have been developed out of these efforts (Ko et al., 2009). These standards have also been applied to clinical pathway modeling before (Scheuerlein et al., 2012). It would be ideal if dynamic checklists support system developments could also benefit from these methodologies. This maps to our secondary research question: "How can we reuse existing modeling languages for the modeling of dynamic checklist support systems?"

The outline of this paper is as follows. Section 2 analyzes the modeling requirements by demonstrating an envisioned use case of our clinical partner. The requirements are used to assess existing modeling languages for clinical processes as well as guidelines. Section 3 derives DCCSS from the requirements. Section 4 demonstrates how DCCSS can be applied to the use case from Section 2. Section 5 compares the differences between our approach and related works. Section 6 elaborates on the support for flexibility in healthcare processes (since BPMN has been critized in this context). Finally, Section 7 concludes the paper.

# 2 BACKGROUND

In this section, we illustrate an envisioned application of checklist to clarify functionality requirements of the checklist model. Then we conclude the modeling requirements for checklist from the envisioned application. Finally, we discuss the criteria of selecting business process modeling languages and clinical rule languages for checklist representation.

## 2.1 Envisioned Application of Checklist

For a better understanding of how a checklist can be used in clinical setting, we describe an envisioned application in a peri-operative period. Lots of efforts on improving quality of care in the peri-operative period by using checklists have been proposed and validated. Researchers have designed checklists for both routine procedures and emergencies in the operating room.

We choose a use case derived from clinical practice to indicate how the checklist can be applied. A patient John Doe is planned for a coronary artery bypass graft (CABG) surgery. Before the operation, a nurse takes his blood for lab tests. The results show that he has a high international normalized ratio (INR), which indicates that he may excessively bleed during the surgery. Then, checklists are sent to the responsible surgeon, anesthesiologist and nurse. The abnormal INR value has been highlighted in the anesthesiologist's checklist as a warning item. Only after these three person confirm every critical items for the operation, the process can continue. At the time the operation is about to start, the operation team stands together to perform a time-out checklist. The operation starts after the time-out checklist is confirmed and signed. During the operation, unfortunately, a cardiac arrest happens. The operation team is under stress as the patient is in danger and warnings are blinking on every monitor. In this situation, they follow a resuscitation checklist, which prioritizes things and guides them out of danger. After the operation finishes successfully, the patient is sent to the intensive care unit (ICU). There, an intensivist and an ICU nurse need to know what happens before and during the operation. Checklists are distributed to these two person to help them to confirm critical patient information. At the same time, they can know from a picture log who did what in which phase. So that if anything goes wrong, they know whom to ask.

## 2.2 Modeling Requirements

From the envisioned approach, we derive a set of requirements capturing the need of representing clinical checklist knowledge comprehensively. There are four modeling components mapping with these four key aspects in checklist application, which are the clinical workflow, the clinical rule, interoperability between workflow and rule, as well as the layout. We conclude the modeling requirements in four items as follow.

**Clinical Workflow.** The dynamic checklist model should support clinical workflow which can be in sequential, parallel or conditional orders and interoperable with other systems.

**Clinical Rule.** The dynamic checklist model should support both simple situational-action rule (SAR) and nested rules, i.e. one rule can be used as the action of another rule.

**Layout.** The dynamic checklist model should support customized layout which facilitates priority for emergency cases.

**Interoperability between Workflow and Rule.** The dynamic checklist model should have the ability to define of relationship between clinical workflow and clinical rule.

## 2.3 Analysis of Existing Modeling Languages

The requirements mentioned previously are partly supported in other mature modeling approaches. These approaches have been used both in the healthcare domain and other industries. For example, clinical pathway management systems are increasingly used in recent years in hospitals. These systems are designed to assign tasks to the right person at the right time. Computerized provider order entry (CPOE) systems can use rules regarding specific patient context like their age, gender and renal function to prevent medical errors. More intelligent guideline based clinical decision support systems use complex clinical rules to help doctors making decisions (De Clercq et al., 2004).

The representing formats used in these systems can be divided into two categories by their design purpose and application domain. One category consists of business process modeling languages, which focuses on describing activities, roles, resources and their relationships in complex business processes. The other category consists of guideline modeling languages which focus on the decomposition of guideline tasks into more detailed clinical decision support rules. The two categories facilitate two aspects of checklist respectively. It would be ideal if we can combine these two formats together.

Business process modeling languages are intended to allow designers to formalize a process definition such as a workflow, in which it can represent in which order activities are carried out and who will perform them. Checklist is a group of activities that should be performed at the proper time by the proper person. Thus checklists have been encoded in such formats. A care path behind a group of checklists are defined as a process. In the process, each checklist is modeled as an activity. An activity is allocated to a role who has a certain function in the whole process. These activities have orders between each other. The simplest order is sequential, which means two activities are performed one by one. To represent this, the concept transition has been used. More complex orders like branching in parallel or conditionally are also supported. In these cases, the concept gateway is used to represent the split and merge of several branches. Most business process modeling tools have graphical model editor to make modeling work easier. Some provide human task user interface designers. Standards from industry have been made to make human tasks sharable between different systems (OASIS, 2012).

Among business process modeling languages, some languages gain more attention from industry and academia, namely Business Process Execution Language (BPEL), Business Process Modeling and Notation (BPMN), XML Process Definition Language (XPDL) and Yet Another Workflow Language (YAWL) (Ko et al., 2009). BPEL is developed for specifying actions within business processes with web services. BPEL uses eXtensible Markup Language (XML) as its representing format which is understandable by IT expert but not domain expert. As an industry standard, BPEL is widely supported in business process management systems. XPDL is designed to interchange business process definitions between different workflow products. It is also written in XML but it has a graphical notation which makes it understandable to domain experts. However, XPDL is designed for interchanging rather than executing. YAWL is an academic language that was designed through a rigorous analysis of workflow patterns. YAWL has graphical notations as well. Different from other industry-driven languages, YAWL is less supported in commercial business process management systems. BPMN is a standard that offers the most expressive and understandable language at the time of writing. The standard also prescribes an interchange format that enables the combination of modeling software and runtime execution software from different industry vendors (OMG, 2011). The BPMN language was designed to be comprehensible by both IT specialists and professionals. Various other authors treat BPMN as a cost efficient, rational, standardized, intuitive and flexible instrument for modeling healthcare processes (Scheuerlein et al., 2012). We conclude the advantages and drawbacks of these languages in Table 1.

While business process modeling communities are developing business process modeling languages, the health informatics community has taken a distinctive approach which is more stressed on decision-making issues. Clinical rules are the main concern of these efforts. In one piece of rule, situation and action

Table 1: Comparison between process modeling languages.

| | IT Expert | Domain Expert | Visualization | Support Systems | Extensibility |
|---|---|---|---|---|---|
| BPEL | + | - | - | + | + |
| BPMN | + | ++ | + | + | + |
| XPDL | + | + | + | - | - |
| YAWL | + | + | - | - | - |

are two essential components. Situation is a group of criteria. When this group of criteria is met, an action will be taken. This action can be to perform a clinical activity or to evaluate another clinical rule. A checklist item or even a checklist can be considered as a specification of an action in these formats. Personalized criteria are modeled as situations to make each checklist (item) specific to patient context.

Different from business process modeling languages, clinical guideline modeling languages concerning decision-making are academia-driven. As a result, these languages are not standardized by international organizations. It makes these languages difficult to be interchanged in different systems. One exception is the Arden Syntax. Arden Syntax has been accepted as an Health Level 7 (HL 7) and American National Standards Institute (ANSI) standard. However, this language has no graphical representation format. This makes Arden difficult to understand for domain experts. Another wide-spread language which takes the advantage of Arden Syntax is GuideLine Interchange Format (GLIF) (Peleg et al., 2001). GLIF is used as the basis of other guideline modeling tools (e.g. EON, SAGE) and has been supported by a commercial rule-based clinical decision support system, Gaston (De Clercq et al., 2004). Mapping between GLIF and other clinical guideline languages is also possible and well studied (Peleg et al., 2001).

Based on our analysis, both BPMN and GLIF have their own advantage for representing checklist knowledge in their domain. From these two languages, we derive our clinical workflow metamodel and clinical rule metamodel respectively.

# 3 THE DCCSS META-MODEL

From the aforementioned analysis of requirements, there are four main concerns while modeling the dynamic checklist, which are the workflow, the rule, the layout and the mapping between workflow and rule. In this section, we present the meta-model concerning these four aspects.

## 3.1 Clinical Workflow

In previous section, we analyzed business process

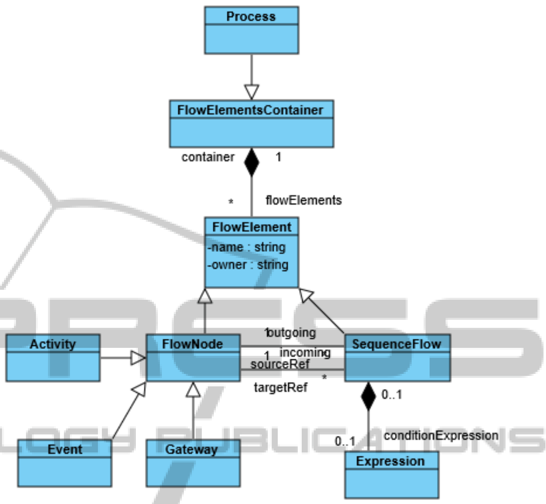


Figure 1: Workflow.

modeling languages and conclude BPMN as the most suitable one for modeling clinical checklist related processes. Our clinical workflow meta-model is derived from BPMN, with the purpose of taking the advantage of BPMN and existing business process management systems as well as their workflow engine.

As shown in Figure 1, the meta-model of clinical workflow abstracts the clinical workflow aspect of the checklist. In the root level, there is a **Process** entity which stands for a clinical process. **Process** contains multiple **Flow Elements**, which stands for constructs in the process. These elements can be divided into two groups. **Flow Node** is the abstraction of activities between system and human. Each node has a participant to specify who is the owner. Specifically, there are three kinds of flow nodes. **Activity** stands for clinical activities that can be performed by human or computer systems. **Event** stands for interactions between current process and other processes or systems. **Gateway** is used for representing branching or merging in the process. A **Sequence Flow** is the link between two nodes.

## 3.2 Clinical Rule

Our clinical rule meta-model is derived from GLIF, which is widely used in healthcare domain. The meta-model of clinical rules shown in Figure 2 is the ab-

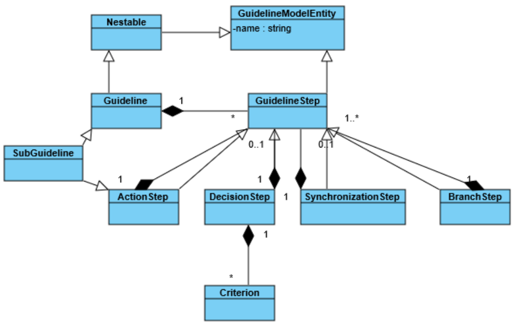


Figure 2: Rule.

stract of situation-action clinical rules that have been used in clinical processes. In the checklist context, all the clinical rules aim to perform an action in specific conditions. This form is very suitable for remainder decision support system like checklist system.

In this meta-model, we use a **Guideline** to represent one clinical rule. Each guideline can have several **GuidelineStep**s. These guideline steps can be divided into four categories, which are **BranchStep**, **SynchronizationStep**, **DecisionStep** and **ActionStep**. Branch step and synchronization step are used to represent parallel decisions. Decision step is used for making decisions based on predefined **Criterion**s. Action step stands for the actions reflect to decisions. For a checklist item, the action is showing the content of the item.

### 3.3 Layout

Checklists is heavily used for intensive care scenarios. In that setting, it is critical for the users to grasp the most important information at the first glance on the checklist. In this sense, layout is an important factor to these checklists.

We describe the meta-model of checklist layout in Figure 3. Every element belongs to a **Page**, which represents one page of checklist. Every element is considered as a kind of **Control**. A control has its width, height, x coordinate, y coordinate and color. There are two categories of controls. One is basic element, which can present text and photo on a page. The other kind is container, in which other controls can be nested.

### 3.4 Checklist

Checklists are presented to users in pages in the correct context. The clinical workflow meta-model answers the question of where and when the user should see the checklist. The clinical rules answer the question of what exactly the user can read in the checklist.

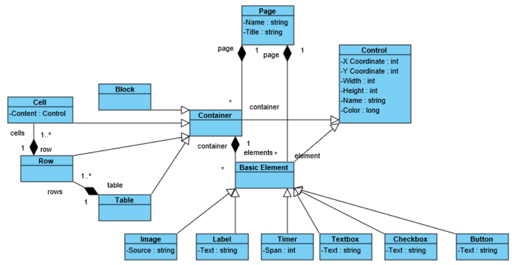


Figure 3: Layout.

However, there remains an unanswered question regarding how the clinical workflow and clinical rules can interact. The interaction between clinical workflow and clinical rules are two-way. On one hand, clinical rules should only be triggered at the right time in the whole process. On the other hand, clinical rules should have the ability to modify the clinical workflow when necessary.

We abstract these requirements in the meta-model shown in Figure 4. Here we define a group of checklist within one process as a three-layer structure. All the checklists are associated with a clinical **Process**. The process has an id to identify itself and a name as a description. Each page of checklist that should be used in certain context is represented as a **Scenario**. Each scenario has an id and name. Besides that, the potential owner attribute is used to specify which user(s) is scheduled to perform the checklist. The phase attribute is used to describe the stage in the whole process. In the real clinical settings, it is possible that people always perform several checklists together as a group, although they are defined separately. Therefore, we design a **Group Scenario** to represent this construct. There is a grouped items attribute in group scenario to indicate which scenarios are in a group. Each scenario can link with several rules before entering and after submission. These rule can effect the workflow itself. In each scenario, there are groups of **Task**s which stand for checklist items in one checklist. The content is represented as description. Moreover, each item is associative with several rules that can be executed before the assignment and submission of the item.

## 4 EXAMPLE

In this section, we demonstrate an example to indicate the applicability of the proposed meta-model. We firstly describe a model of an example checklist in the selected format. Then, we discuss a prototype system based on our meta-model. The system has an interface with an EMR system and it can execute the ex-

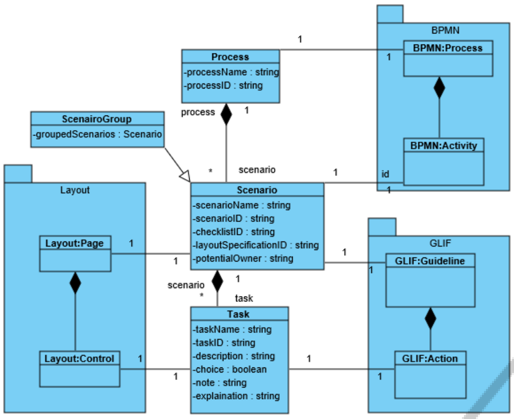


Figure 4: Checklist.

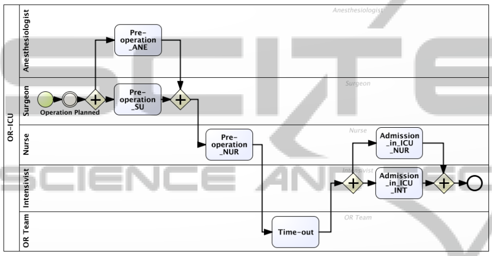

Figure 5: Peri-operative Workflow.

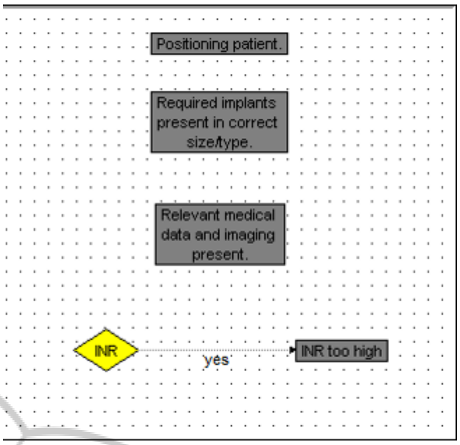


Figure 6: Clinical Rule Example.

```xml
<Process processID="ORICU_v2" processName="ORICU_v2">
  <Scenario scenarioID="Gaston_12" scenarioName="Timeout_in_OR_SU" />
  <Scenario scenarioID="Gaston_9" scenarioName="Timeout_in_OR_ANE" />
  <Scenario scenarioID="Gaston_15" scenarioName="Timeout_in_OR_NUR" />
  <Scenario scenarioID="Gaston_21" scenarioName="Post-operative-checks_SU" />
  <Scenario scenarioID="Gaston_18" scenarioName="Post-operative-checks_ANE" />
  <Scenario scenarioID="Gaston_24" scenarioName="Post-operative-checks_NUR" />
  <Scenario scenarioID="Gaston_27" scenarioName="Admission_ICU_INT" />
  <Scenario scenarioID="Gaston_30" scenarioName="Admission_ICU_NUR" />
</Process>
```

Figure 7: XML Example.

ample checklist model effectively.

## 4.1 A Checklist Model

In this section we model the use case from Section 2.1 utilizing our proposed meta-model.

Firstly we model the clinical process in the BPMN language (Figure 5). We created five swim lanes in the BPMN model to represent the five participants. We abstracted eight steps which need to perform checklists as eight tasks in the model. The pre-operative checks are done by three roles in parallel. So we use parallel gateways to represent this kind of relationship.

Then we model checklist items in each checklists in GLIF (Figure 6). Each checklist is considered as a guideline in GLIF. Each item in the checklist is considered as a clinical rule, which should contain at least an action that indicates the specific task for the user. Decisions are optional to a rule. If there is no decision in one item, it means the action will be executed by default. However, decisions are important if we want patient context specific items (e.g., checking the INR rule). In that case, we express that in an GLIF expression.

After that, we need to indicate how the workflow elements should link with checklists. We use an XML file to represent these relationships (Figure 7). Every checklist is nested in the process. The scenario ID attribute is used to identify which checklist in the GLIF model should be linked.

In the end, we define the layout of the checklist in HTML 5 format (Figure 8). All the controls are mapped to HTML 5 types. The relationship between a checklist and its layout is also defined in the previously mentioned XML model.

## 4.2 A System Based on the Meta-model

Tracebook is a prototypical system for executing DCCSS models (Nan et al., 2014). Based on the DCCSS structure, Tracebook has four main components (in Figure 9). The workflow engine is designed to support the workflow model. In this part we interfaced with BizAgi Express business management system's APIs[1]. The rule engine is used to support clinical rules. In this part, we interfaced with Gaston (De Clercq et al., 2004), which is a GLIF 2.0 based clinical rule engine. We implemented the checklist engine to deal with the interoperability between the workflow engine and rule engine based on our mapping model. The checklist engine also send checklists to the UI renderer in XML format. The UI renderer uses XSLT to interpret the XML into HTML 5 format and show them to users as checklists. This system has been implemented and under evaluation in

[1]See http://wiki.bizagi.com/.

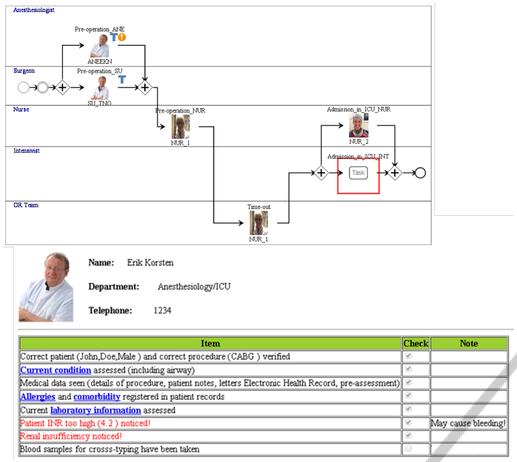


Figure 8: Page Layout.

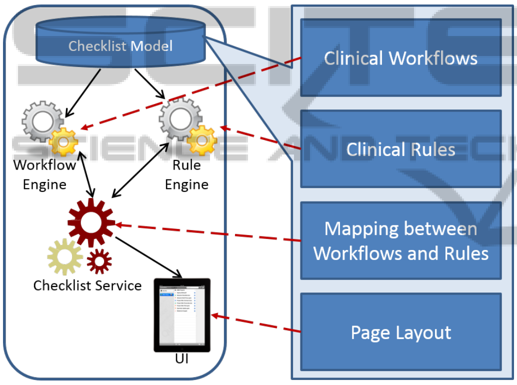


Figure 9: System Framework.

Catharina hospital in Eindhoven, the Netherlands. It interfaces with the hospital EMR system.

## 5 RELATED WORKS

In recent years, other researchers have also contributed to the field of checklist modeling.

Färber et al. proposed to a method to model clinical processes, where they propose checklist as a new type construct in the clinical process modeling (Färber et al., 2007). In their model, they treat activities strongly belonging together to one critical execution phase and without execution order as a checklist. Their model mainly focuses on how to simplify the modeling of clinical processes rather than enabling the representation of a clinical checklist. Thus, that line of work did not yet consider how to model clinical rules nor how to integrate layout aspects.

Avrunin et al. proposed smart checklists for human-intensive medical systems (Avrunin et al., 2012). They use a process modeling language called Little-JIL. Little-JIL supports task assignment and complex orderings of tasks. Also, checklist items can change dynamically during the execution, driven by changes of the context. There are several differences between (Avrunin et al., 2012) and our approach. Firstly, our work creates a general modeling framework for checklists which aim to reuse existing modeling languages, whereas their approach focuses on utilizing a specific modeling language mainly for clinical process. Secondly, checklist layouts and groupings of checklists were not yet considered by Avrunin et al.

## 6 DISCUSSION

Healthcare processes are normally highly dynamic and ad-hoc. The order of activities might be changed during the execution according to patient status. Some activities can be removed or added during the execution. These dynamic features are difficult to be represented in procedure process models. Case handling or case management is proposed to represent the dynamicity. Instead of activities or routines, case handling focuses on the concept case, which is specific to a certain context. Case Management Modeling and Notation (CMMN) is proposed to standardize the case handling model (OMG, 2014). Different with BPMN, CMMN takes a declarative approach rather than a procedural approach. CMMN aims to represent what activities a person *can* do, whereas BPMN aims to represent how activities *should* be performed by a person.

The CMMN can be more representative in describing care-givers' behavior as they normally do. However, we argue that the use of checklist is to *standardize* practitioners' behavior at the very moment, rather than enable them doing critical thing in *ad-hoc* To standardize the behavior, a procedural description is more clear and easy to build. It is also important to show the status of the whole process to users in the run time. This meets the design target of BPMN. Therefore, we have chosen BPMN as our process definition language. Additionally, ad-hoc checklists (e.g., the aforementioned resuscitation checklist) can also benefit from the event handling mechanisms in BPMN.

Data aspects are also important to make clinical decision support systems portable to different healthcare organizations using heterogeneous data source. Currently, our approach benefits from the underlining data model in GLIF, which is based on the Reference Information Model. One limitation is that the workflow engine cannot benefit from this data model yet. This is not preventing us from performing initial clinical evaluations since our workflow models only use limited amount of data (which can be handled with code supplements to the DCCSS model). However, defining an explicit and independent data model which is sharable between all the four DCCSS aspects is an important line of ongoing work.

## 7 CONCLUSION

In this paper, we propose DCCSS as a meta-model for dynamic checklist, which reuses existing infrastructure from outside the clinical domain. By analyzing a use case derived from clinical practice, we summarize four checklist modeling requirements, which are the clinical workflow, the clinical rule, the layout and the interoperability between the workflow and the rule. According to these modeling requirements, we have reviewed the popular business process modeling languages and clinical rule modeling languages and we have chosen BPMN and GLIF as the best building blocks for DCCSS. By then adding specific constructs which were not yet supported by the more general BPMN and GLIF metamodels, we have ensured that DCCSS meets checklist-specific modeling requirements. Finally, we have modeled a peri-operative checklist to demonstrate the effectiveness of the meta-model. The DCCSS model can be executed on a prototypical system Tracebook which can interface the EMR system of our clinical partner.

One advantage of our approach is that it gives a platform independent model which is not tied to a specific workflow engine or rule engine product. This is important since hospitals already have a variety of process aware information systems (which contain workflow engines) and decision support systems (which have rule engines). When using small wrapper systems such as Tracebook, hospitals can easily reuse their existing products and knowledge for realizing dynamic checklist functionality. When the hospital legacy systems lack support for standards then the integration work will still be significant but the efforts of building a standard interface on top of a proprietary legacy system are orthogonal to building dynamic checklist support on top of that.

In ongoing work, we are extending DCCSS with support for platform independent data access from each of its four perspectives.

## REFERENCES

Avrunin, G. S., Clarke, L. a., Osterweil, L. J., Goldman, J. M., and Rausch, T. (2012). Smart checklists for human-intensive medical systems. In *IEEE/IFIP International Conference on Dependable Systems and Networks Workshops (DSN 2012)*, pages 1–6.

Babayan, R. K. (2011). Effect of a comprehensive surgical safety system on patient outcomes. *The Journal of urology*, 185(4):1329–30.

De Clercq, P. a., Blom, J. a., Korsten, H. H. M., and Hasman, A. (2004). Approaches for creating computer-interpretable guidelines that facilitate decision support. *Artificial intelligence in medicine*, 31(1):1–27.

Färber, M., Jablonski, S., and Schneider, T. (2007). A comprehensive modeling language for clinical processes. In *European Conference on eHealth 2007, GI, Oldenburg, Germany. Lecture Notes in Informatics (LNI)*, pages 77–88.

Ko, R., Lee, S., and Lee, E. (2009). Business process management (BPM) standards: a survey. *Business Process Management Journal*, 15(5):744–791.

Nan, S., Van Gorp, P., Korsten, H., Vdovjak, R., Kaymak, U., Lu, X., and Duan, H. (2014). Tracebook: A dynamic checklist support system. In *Computer-Based Medical Systems (CBMS), 2014 IEEE 27th International Symposium on*, pages 48–51.

OASIS (2012). Web Services Human Task (WS-HumanTask) Specification Version 1.1. [online] Available at: http://docs.oasis-open.org/bpel4people/ws-humantask-1.1.html. [Accessed 09 September 2014].

OMG (2011). BUSINESS PROCESS MODEL AND NOTATION (BPMN). [online] Available at: http://www.omg.org/cgi-bin/doc?dtc/10-06-04.pdf. [Accessed 09 September 2014].

OMG (2014). Case Management Model and Notation. [online] Available at: http://www.omg.org/spec/CMMN/1.0/PDF/. [Accessed 09-September-2014.

Peleg, M., Boxwala, A. A., Bernstam, E., Tu, S., Greenes, R. A., and Shortliffe, E. H. (2001). Sharable representation of clinical guidelines in glif: relationship to the arden syntax. *Journal of biomedical informatics*, 34(3):170–181.

Scheuerlein, H., Rauchfuss, F., Dittmar, Y., Molle, R., Lehmann, T., Pienkos, N., and Settmacher, U. (2012). New methods for clinical pathwaysbusiness process modeling notation (bpmn) and tangible business process modeling (t. bpm). *Langenbeck's Archives of Surgery*, 397(5):755–761.

Weiser, T. G., Haynes, A. B., Lashoher, A., Dziekan, G., Boorman, D. J., Berry, W. R., and Gawande, A. a. (2010). Perspectives in quality: designing the WHO Surgical Safety Checklist. *International journal for quality in health care*, 22(5):365–70.