Process Driven Guidance for Complex Surgical Procedures

George S. Avrunin[1], Stefan C. Christov[2], Lori A. Clarke[1], Heather M. Conboy[1], Leon J. Osterweil[1], Marco A. Zenati[3]
[1]University of Massachusetts, Amherst, MA, USA; [2]Quinnipiac University, Hamden, CT, USA; [3]Harvard Medical School, Boston, MA, USA

Abstract

Surgical team processes are known to be complex and error prone. This paper describes an approach that uses a detailed, validated model of a medical process to provide the clinicians who carry out that complex process with offline and online guidance to help reduce errors. Offline guidance is in the form of a hypertext document describing all the ways the process can be carried out. Online guidance is in the form of a context-sensitive and continually updated electronic "checklist" that lists next steps and needed resources, as well as completed steps. In earlier work, we focused on providing such guidance for single-clinician or single-team processes. This paper describes guiding the collaboration of multiple teams of clinicians through complex processes with significant concurrency, complicated exception handling, and precise and timely communication. We illustrate this approach by applying it to a highly complex, high risk subprocess of cardiac surgery.

1 Introduction

Of the patients who undergo cardiac surgery each year, 28,000 will have at least one adverse event and one-third of the deaths associated with coronary artery bypass graft (CABG) operations may be preventable [1]. To address these concerns, we are developing approaches aimed at reducing preventable procedural errors and thereby improving patient outcomes.

Specifically, we are developing an approach for modeling complex processes, validating those process models to detect defects, and then using those validated process models to automatically provide offline training and online guidance to help clinicians while carrying out complex and error prone medical procedures. Previous papers have described the process modeling language [2], the process validation approaches used, and case studies of their application to medical processes (e.g., [3, 4]). For instance, the chemotherapy process case study reported a 70% reduction in the number of errors reaching the patient [4]. Those medical processes studied mostly involved only one or a few clinicians working in close concert. In our current project, we are applying this approach to cardiac surgery processes, a setting where multiple teams (surgeons, nurses, anesthesiologists, and perfusionists) must interact with each other and with a variety of devices in real time in order to successfully complete their tasks. Unfortunately cardiac surgery processes have been reported to average four errors an hour during the four or more hours of surgery [5]. This paper focuses on how our process-driven guidance approach provides support for such teams by providing them with process insights via two views: a *Narration View* that includes a description of the process in natural language and a *Smart Checklist View* that uses monitored events and other contextual information to provide online guidance as the process is being executed. We illustrate our approach with an example from CABG surgery where the cardiopulmonary bypass (CPB) pump (also known as the heart-lung machine) is used.

The next section of this paper describes our approach in more detail, including the elicitation and validation of our process models, the Narration, and the Smart Checklist. Section 3 discusses related work, and Section 4 describes the current status of this project, evaluation efforts, and plans for future work. We illustrate our capabilities by demonstrating how they can be used to support the **wean from bypass** subprocess, a particularly critical, yet intricate part of CABG surgery, when the patient, who has been on bypass with the CPB pump, must be separated from bypass.

2 Approach

A key element in our approach is a carefully elicited and validated model of the surgery team process. This detailed and rigorously defined model captures how people and devices carry out the process, including the normative situations as well as non-normative (exceptional) situations that can arise and how they are handled. As discussed below, we use the Little-JIL process modeling language [2], because it supports the specification of such complex processes and has

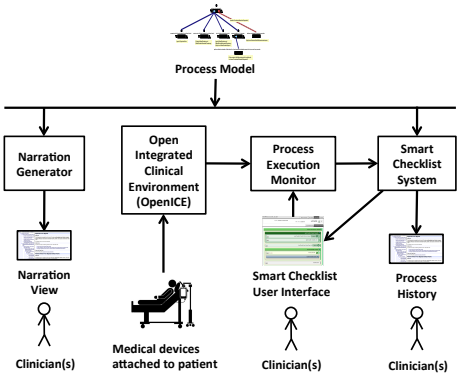

Figure 1: Basic architecture of the process guidance system

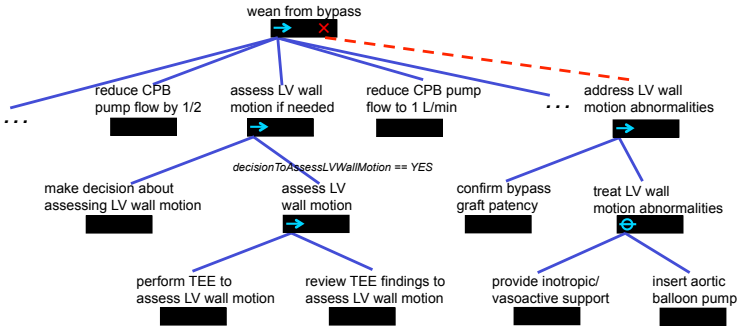

Figure 2: A portion of the **wean from bypass** subprocess written in Little-JIL

well-defined execution semantics that enable automated validation and online guidance.

Figure 1 shows the architecture of our process guidance system. A carefully elicited, detailed, rigorously defined, and validated model of the process, shown at the top, is the basis for the guidance provided by the system. The left side of Figure 1 depicts the *Narration Generator*, which produces a precisely phrased natural language *Narration View* of the process, a hypertext document that presents such features as the process hierarchical structure, the details of each step, and the flow of control and data through the process. This static description, explained in more detail in Section 2.2, documents all the potential paths through the process model (representing possible executions of the process), and provides an electronic process guide that can be used for reference, training, and validation of the model.

The *Smart Checklist System*, shown on the right in Figure 1 and discussed in Section 2.3, creates a dynamically-updated, context-aware user interface to guide process performers as they execute their steps. This system is designed to receive information about the occurrence of process events from human process performers and from the Open Integrated Clinical Environment (OpenICE) [6], also shown in Figure 1. OpenICE receives data from medical devices such as the CPB pump, the lung ventilator, and the hospital's electronic medical record application. The system informs process performers about the history and state of the process execution, including what other process performers are doing and what steps are pending, updating its displays as events occur and the process execution state changes.

The Little-JIL editor and interpreter are fully implemented, as are the Narration Generator and the Single-User Smart Checklist. The Multi-User Smart Checklist is currently in the late stage of development. For this work, the computer scientists collaborated with a cardiac surgery team from the Veterans Affairs Boston Healthcare System (VABHS).

2.1 Process Elicitation, Modeling, and Analysis

Our process models are elicited, and iteratively improved, by interviewing domain experts, consulting the medical literature, and codifying best practices. We believe that a sufficiently detailed model must specify activities, artifacts, agents, and resources. Activity specification requires that process steps be specified in sufficient detail, typically accomplished through support for hierarchical decomposition. It also requires support for procedural control flow constructs including choice, iteration, concurrency, and multi-level, scoped exception handling. Conciseness and comprehensibility are also important, and are greatly facilitated by support for procedural abstraction. Specifying the structure of artifacts, such as physical objects, data items, and messages, and how they are created and used by process performers is also a key part of modeling a process. Agents are the humans, devices, and software applications that actually perform the process steps. Specifying agents entails representing aspects such as agent capabilities, capacity, availability, and substitutability. Step performance may also require other resources, such as harvested blood vessels, and these must be specified as well. Since there are often complex interactions among the activities, artifacts, agents, and resources, we need the process models to have well-defined execution semantics to support automated reasoning for validation purposes and an execution engine to appropriately update the online guidance.

Peleg et al. surveyed medical guideline modeling languages, focusing on domain-specific languages [7]. Other researchers have modeled processes using general-purpose languages such as BPMN [8]. Both domain-specific and general-purpose languages often cannot express some of the complexity of the surgery team processes, in particular resource specification, concurrency, and exception management. Many of these languages have informally defined semantics and hence do not support automated reasoning or an execution engine. For this work, we use the Little-JIL process modeling language, which is both expressive and has well-defined execution semantics.

Figure 2, a visualization of part of the Little-JIL model of the **wean from bypass** subprocess, illustrates some salient features of our process modeling approach. Little-JIL steps are depicted as step bars, black rectangles whose names appear as text above them. Solid blue lines below a step bar point to substeps that comprise the step's decomposition. Thus, **reduce CPB pump flow by 1/2**, **assess LV wall motion if needed**, and **reduce CPB pump flow to 1 L/min** are all substeps of **wean from bypass**. The right-facing arrow in **wean from bypass** indicates that these substeps are to be performed in left-to-right order. Thus, **assess LV wall motion if needed** must occur only after the CPB pump flow has been halved, and must precede further reducing the pump flow to 1 L/min. Note that there are some steps that precede the first reduction in flow, and others that follow the second. These have been elided because of space limitations, and are indicated here by ellipses, which are not part of the Little-JIL language. The step **assess LV wall motion if needed** is itself further decomposed into a step where the step's agents make a decision about whether to do an assessment of LV wall motion, followed by a step for doing the assessment. Note that the edge leading into this latter step, **assess LV wall motion**, is annotated with a logical proposition that uses an artifact generated by the prior step to determine whether the substep is to be performed. The **assess LV wall motion** step is further decomposed into two substeps that specify that first the assessment is done using TEE, and that next the assessment is reviewed.

Little-JIL facilitates specifying that a step may throw an exception when a non-normative situation has arisen, how to handle that situation, and how to continue after handling it. In this case the assessment may indicate abnormalities in the motion of the LV wall, and the step definition accordingly specifies that the step performance may cause the LV wall motion abnormality exception. This results in a search up the step decomposition structure, looking for a handler for this exception. Exception handlers may be attached to any step, and are indicated by a large red X on the right of the step bar, and a dotted red line leading downward to a step that is a specification of the process for handling the exception. In this case the handler, **address LV wall motion abnormalities**, is a part of the **wean from bypass** step, and is itself a structure of substeps. Of particular note is **treat LV wall motion abnormalities**, a step having a slashed O on the left of its step bar. This indicates that its substeps are alternatives, only one of which is to be performed. This choice is made by the agent specified as the performer of the step (the specification is not shown in the figure). Note that an = icon can also be used to indicate when substeps may be performed in parallel, although this icon is not used in the part of the process shown in the figure. A full description of the Little-JIL language can be found in [2].

Elicitation begins with observing and interviewing domain experts and process performers using various methods [9] and consulting available process documentation to construct an initial model. (In our current project, we are in the process of recording video and audio from 36 cardiac surgeries and annotating the recordings. These recordings will also be used to refine the model.) We then return to the process experts and review the model with them. Typically even expert process performers are not sure of some aspects of the process, especially those involving exceptional situations, and our model elicitation efforts encourage process performers and designers to consider cases where the process itself needs more work. All of this necessitates modifications to the model, further review with the domain experts, and continued iteration. This process modeling effort is often time consuming. For instance, the cardiac surgery process modeling effort has been spread over the last two years, including developing high-level models of the CABG and aortic value replacement processes and then elaborating several key shared subprocesses. For each (sub)process, we worked for one to two months and usually held weekly elicitation/review meetings for 30 to 90 minutes to gather feedback. After each meeting, a process modeling expert then carefully modified the (sub)process models to address the feedback. This effort, however, is worthwhile because a process model can be leveraged by both the automated validation and guidance approaches to try to reduce errors. That model can also be shared with other healthcare facilities and customized for them if needed. For instance, the CABG model uses the VABHS team's terminology but could be customized to use standardized terminology such as that provided by SNOMED CT [10].

Little-JIL has well-defined semantics and, thus, supports powerful reasoning integral to our iterative approach to

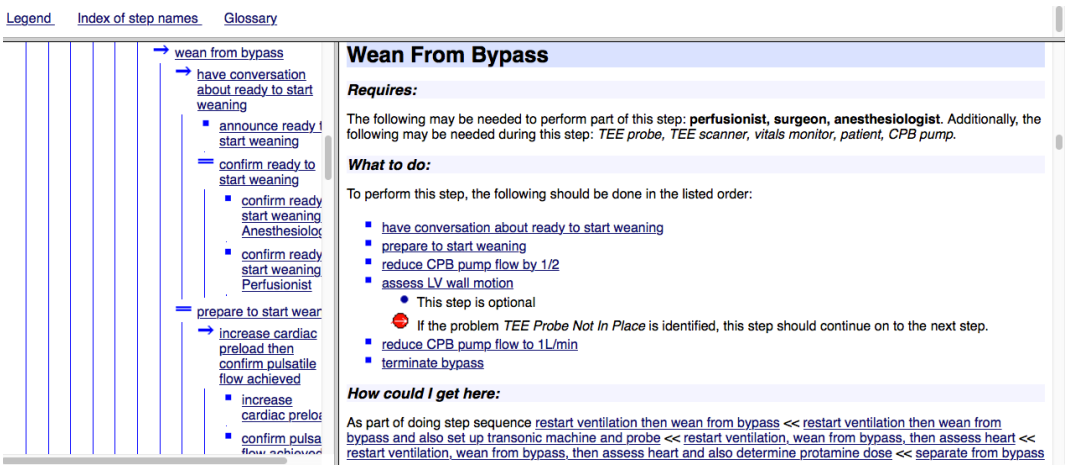

Figure 3: A portion of the Narration for the **wean from bypass** subprocess

assuring the correctness of our models. Once we define a model that the process experts agree covers the necessary behavior, that model is further validated via model checking and fault tree analysis [3]. Model checking can determine if there is a path through the process model whose execution can violate a specified requirement, such as one stating that the CPB pump and the lung ventilator cannot both be off at a particular point in the process. Fault tree analysis provides descriptions of how unwanted situations (such as the pump and the ventilator both being off) could arise from the incorrect performance of some combination of process steps. The Little-JIL semantics are used to automatically translate to the input formalisms of these static analysis tools. (These semantics are also used to implement a Little-JIL interpreter employed to appropriately update the process execution state described in Section 2.3.) When our automated tools initially find problems with a model, it is typically because the model is incomplete or does not correctly describe the process. This leads to modification of the model in consultation with domain experts and revalidation of the modified model. After further iteration, problems with the process itself begin to emerge.

2.2 Narration Generator

Although some of our medical collaborators have quickly learned to read and understand Little-JIL models, Little-JIL has some subtle features that can be confusing to non-experts. Moreover, because Little-JIL steps (which can represent a single activity or an entire subprocess) are like programming language procedures, references to them can be far from their definitions in the visual representation of a Little-JIL model, making it difficult to trace some possible paths through a process model. We address these problems by providing an automated Narration Generator that produces a hypertext "Narration" of a Little-JIL model. The Narration allows users to trace paths through the model and to browse the model to explore different process scenarios.

Figure 3 shows a Narration of part of **wean from bypass**. The Narration has two main panels: a table of contents on the left and detailed descriptions of process steps on the right. The table of contents lists the names of the process model steps with icons representing the steps' control flow (e.g., sequential, parallel, choice) and uses indentation to show the hierarchical structure of the Little-JIL model. The table of contents also shows the exception handlers (denoted by red circles), along with their continuation semantics (denoted by different arrows in the red circles). Clicking on a step name in the table of contents brings up the detailed description of that step in the right-hand panel.

A step description gives various attributes of the step, including the step's name (e.g., **Wean from Bypass**), and a short paragraph listing the agents who perform the subprocess represented by the step (e.g., anesthesiologist, perfusionist, and surgeon for the parent step, **Wean from Bypass** step) and other resources required to perform the step (e.g., vitals monitor, a TEE probe, and the CPB pump, and TEE scanner). Another part of a step's description is a specification of what is required in order to successfully complete that step. For example, to successfully complete the step **Wean from Bypass** in Figure 3, the several listed substeps, starting with **have conversation about ready to start weaning**, must be performed in order. Other steps may have far more complex completion requirements.

The step description also contains information about potential problems that might arise when one of the step's substeps is performed and specification of how they should be handled. An example appears indented under the step **assess LV wall motion if needed**.

To help the user of the Narration place the described step in the context of a process execution, the step description section ends with information about what sequence of process steps could lead to the described step (the "How could I get here" section in Figure 3). Other information that could be included in the step description section, but is not shown in Figure 3, is whether the step is part of an exception handler as well as any special conditions that must be satisfied before the step can be started or before it is completed.

Clicking on the "Legend" link at the top of the Narration pops up a window explaining all the icons used in the Narration. The other links next to "Legend" provide access to a listing of all the steps in alphabetical order and a glossary of terms used in the Narration. When using the Narration to follow a possible process path, the user who gets to **Wean from Bypass** could read the information about the step and then click on the substep **have conversation about ready to start weaning** in either the table of contents or the description of **Wean from Bypass**, discovering that this substep involves the surgeon making an announcement, followed by both the anesthesiologist and perfusionist confirming that they are ready. The user could then see the details of the next substep by clicking on **prepare to start weaning** in the table of contents or by clicking on **wean from bypass** in the "How could I get here" section of the description of **have conversation about ready to start weaning** to return to the description of **Wean from Bypass** and then clicking on **prepare to start weaning** in the list of substeps of **wean from bypass**.

The Narration Generator is template-driven. It takes a Little-JIL process model and a set of phrasing templates, which are parameterized, natural language phrases that correspond to the different semantic features of the Little-JIL language, possibly supplemented by some customization rules that determine what to show, and weaves them into an XML document. A set of formatting templates are then used to produce formatted output. At this time, we only produce an HTML document, but the system is designed so that other formatting templates could generate other suitable documents, such as ebooks, from the same XML file.

2.3 Smart Checklist System

As described earlier, the Smart Checklist System uses the process model and information about the ongoing process execution obtained from process performers and medical devices to provide continually updated guidance to the multiple teams involved in the process. The Smart Checklist user interface shows past steps (the execution history so far); steps currently being performed, including decisions that must be made; and a short look ahead at future steps that are still possible until additional decisions are made. It also displays some (user selectable) information about the state of key devices. The user interface can show this information from the perspective of each team. The user can select which perspectives to display and the Smart Checklist System keeps those perspectives appropriately synchronized. In particular, the user interface indicates when a team is performing process steps independently, collaborating with other teams to perform a step, or waiting for another team to finish a step to continue. The system supports multiple instances of the user interface, configured separately, so each team could have its own display, perhaps together with a single large display showing all the perspectives.

The *Process Execution Monitor* is responsible for gathering a stream of real-time process execution events. Each *process execution event* needs to store information about the step performed, including information about which human or automated agent performed that step, any resources or artifacts used or created by the step, any exceptions thrown, and a timestamp of when the step was completed. The stream of process execution events forms the process history. For now, human process performers, or human scribes, can use the Smart Checklist user interface to record such process execution events. To reduce the amount of human input required, we are currently integrating OpenICE, an open-source implementation of the ASTM F2761-09(13) standard [6], to be able to register with the electronic medical record application and medical devices to automatically receive their process execution events.

Figure 4 shows an initial portion of such a context-aware, dynamic Smart Checklist to guide the overall team through the **CABG** process. Specifically, this figure shows the overall team, consisting of the Anesthesiology (denoted 'A'), Nursing (denoted 'N'), Perfusion (denoted 'P'), and Surgery (denoted 'S') teams, working collaboratively on the **Sep-**

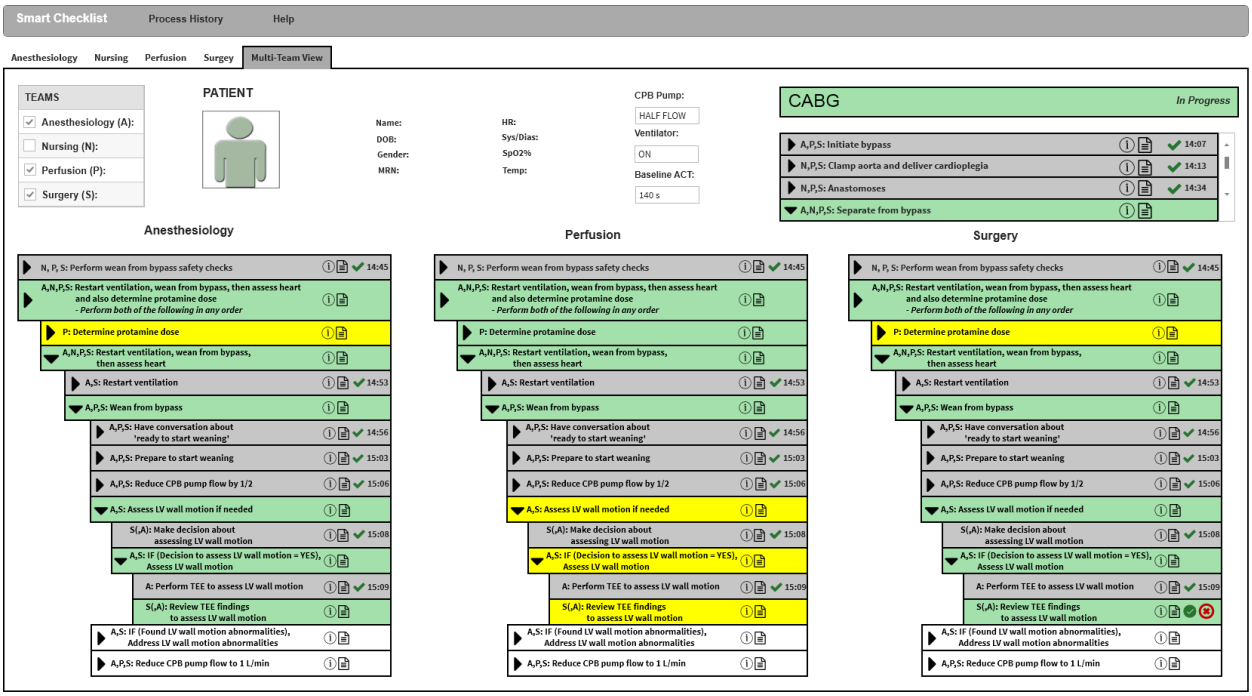

Figure 4: A mock up of the initial portion of the Smart Checklist for the **wean from bypass** subprocess where the surgeon, perhaps with help from the anesthesiologist, has started the **review TEE findings to assess LV wall motion** step

arate from bypass subprocess. The "Process History" menu can be used to generate the documentation of the process executed so far. The "Help" menu brings up a Legend to improve checklist understanding as well as Preferences to customize the information shown in the Checklist and how it is displayed. Our Checklist supports context-awareness along several dimensions including the team information (shown at the top left by the team tabs and table), the patient information (shown at the top middle), and the process execution state (shown at the top right as well as the bottom).

In Figure 4, the user of this display decided to show a Multi-Team View (by selecting the team tab labeled with that view). This user could consult the team table (shown at the top left) to get such information as each team member's full name, specialty team, and expertise level. The patient information (shown in the top middle) can be customized to display selected patient identifiers (shown at the left), real-time vital sign data (shown in the middle) provided by OpenICE, and key process data such as the patient's *baseline ACT* that should be achieved by the end of the **Separate from bypass** subprocess. The process execution state includes the high-level subprocess listing (shown on the top right) that shows that the **Anastomoses** subprocess has been completed (denoted by the gray background and green checkmark with a timestamp) and the **Separate from bypass** subprocess that is currently being performed (denoted by the green background).The process execution state also includes the current subprocess listing (shown at the bottom), in this case the **Separate from bypass** subprocess listing. Because the user decided to focus on the complex interactions among the anesthesiology, perfusion, and surgery teams (by selecting their items in the team table), the **Separate from bypass** subprocess listing consists of the perspectives for anesthesiology (shown at the bottom left), perfusion (shown at the bottom middle), and surgery teams (shown at the bottom right). The indentation in each perspective reflects the step hierarchy in the process model. The process listings can also be customized to select which hierarchical steps should be shown and how to display the steps.

Each of the three perspectives shows that the nursing, perfusion, and surgery teams completed the past step **Perform wean from bypass safety checks** (denoted by the gray background and the green checkmark with its timestamp '14:45'). All four teams are now performing the current step **Restart ventilation, wean from bypass, then assess heart and also determine protamine dose** (denoted by the green background). In the perspective for the surgery

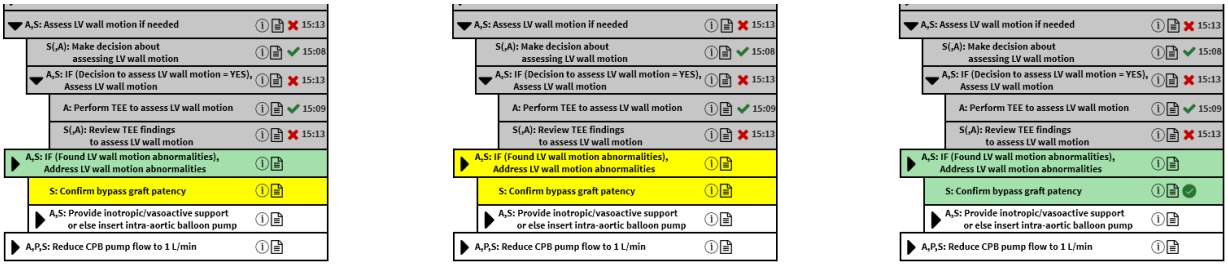

Figure 5: The updated Checklist after the surgeon (and anesthesiologist) reviews TEE findings to assess LV wall motion and reports the exception *Found LV wall motion abnormalities*

team, this current step specifies that the perfusion team is working on the step **Determine protamine dose** (shown in yellow because the surgery team must wait for the perfusion team to complete this step) while at the same time all four teams are working together on the step **Restart ventilation, wean from bypass, then assess heart** (shown in green for the surgery team as well as the anesthesiology and perfusion teams). Specifically, the anesthesiology, perfusion, and surgery teams are currently working together on step **Wean from bypass** (shown with a green background in all three perspectives' views) and already completed the first three substeps.[1] Because the anesthesiology and surgery teams decided to assess the LV wall motion, the anesthesiology team already completed step **Perform TEE to assess LV wall motion** (denoted by the gray background and the green checkmark with its timestamp '15:09'). The surgery team is working on the current step **Review TEE findings to assess LV wall motion** (denoted by the green background as well as the checkmark and X buttons). This team may consult the anesthesiologist about this review (denoted in the anesthesiology team's perspective by the green background but no buttons). On the other hand, the perfusion team's perspective shows that the perfusionist needs to do step **Determine protamine dose**. The perfusionist is also on standby waiting for the anesthesiology and surgery teams to complete current step **Assess LV wall motion if needed** (shown with a yellow background) before the perfusionist may start future step **Reduce CPB pump to 1 L/min** (shown with a white background).

The surgeon, perhaps with the help of the anesthesiologist, must use this review of the TEE findings to decide if any LV wall motion abnormalities are found. For a normative execution where no such abnormalities are found, the surgeon should click the green checkmark button. This execution will continue on to the future step **Reduce CPB pump flow to 1 L/min**. For an exceptional execution where these abnormalities are found, the surgeon should click the red X button to bring up a dialog box to report the problem *Found LV wall motion abnormalities*. This execution first addresses the problem by performing future step **If (Found LV wall motion abnormalities, Address LV wall motion abnormalities** and then continues on to the future step **Reduce CPB pump flow to 1 L/min**. The surgeon may also click on the note button if they want to bring up a dialog box to type in their clinical notes about the review.

Figure 5 shows updates to the Checklist shown in Figure 4 after the surgeon, perhaps after consulting the anesthesiologist, completes step **Review TEE findings to assess LV wall motion** and reports the problem *Found LV wall motion abnormalities* (denoted by the gray background and the red X with its timestamp '15:13'). Based on the exception management specified in the **Isolated CABG** process model, this also causes the steps **If (Decision to assess LV wall motion = YES), Assess LV wall motion** and **Assess LV wall motion if needed** to complete and report the identified problem (denoted by the gray backgrounds and red Xs with their timestamps). For this exceptional situation, the anesthesiology and surgery teams start the current step **If (Found LV wall motion abnormalities), Address LV wall motion abnormalities**. This current step specifies that the surgery team first must do current step **Confirm bypass graft patency** (shown with a green background and green checkmark button) and then the anesthesiology and surgery teams must do the future step **Provide inotropic/vasoactive support or else insert intra-aortic balloon pump**.

The Smart Checklist System can be used to produce Process History documentation. If the process is still in progress, this documentation describes the process execution history seen so far. If the process is completed, the documentation describes the actual path through the process that was executed, which could be used as post-procedure documentation. The Process History documentation consists of a summary of the particular path and a listing of the steps in that path.

[1]The patient information shows that the ventilator is 'ON' and the CPB pump is at 'HALF FLOW.'

The summary includes information such as the team members responsible for performing the process and some of the patient data such as their identifiers and vital signs collected. As noted earlier, a step's documentation includes the team member(s) responsible for performing that step, any data required/produced by the step (including any clinical notes), any exceptional situations that occurred during that step, and the timestamp for when the step was completed. The Process History Documentation Generator works in a similar manner to the Narration Generator. It takes as input the path through the process and produces an XML document to describe that path. The generator then applies templates to that XML document to produce the Process History documentation in various formats such as a table.

3 Related Work

There have been a number of approaches developed for providing process cognitive aids, including various forms of checklists (e.g., [11]) and process guides (e.g., [12]). Paper checklists have been shown in a number of studies to significantly reduce errors in medicine (e.g., [11]). One objection to such checklists, however, is that they cannot represent all of the complex aspects of the processes such as parallelism, team communication, and exception management. Another related objection is that they are static and, hence, unable to make adjustments for context.

Electronic checklists have been introduced to visualize the process execution state and dynamically update that state based on the actual process execution events executed, either by medical team members or by automated components. Some of these electronic checklists provide simple visualizations of paper-based checklists. Others support complex visualizations of the process execution state, including information about the medical team, the patient, and process steps. Some of this work is also based on a process model, most notably the Tracebook project [13]. While these process-model based projects share our goals, the process notations that they use seem to lack the powerful semantics needed (e.g exception management, resource specification, and procedural abstraction) to represent the details and precision required for complex processes such as cardiac surgery.

Surgical process models (e.g., [14]) that focus on the activities of the surgeon, typically at the level of describing which hand is used to apply what instrument to what anatomical structure, provide a somewhat different perspective. The contexts provided by our higher-level team activities might be useful in helping to sharpen and adapt these models of fine-grained activities; something we look forward to investigating in the future.

4 Discussion and Future Work

Although the focus of considerable research, healthcare safety remains a significant concern. Checklists have shown promise in several medical settings, but both paper and electronic checklists have significant drawbacks, especially in complex, team-based, and exception-rich healthcare settings such as cardiac surgery. Our project is developing a Smart Checklist to address many of these drawbacks.

Building on a precisely defined process model that captures concurrency, synchronization, and scoped exception handling, our system offers a variety of guidance. The offline Narration provides a hypertext natural language description of the process, encompassing all possible ways the process can be executed. A user can step through a hypothetical execution of the process, examining detailed descriptions of the steps, or can move forward and backward along different potential executions. This electronic process guide can be used by novices during training or by experts contemplating process improvements. In this latter case, the associated analysis techniques (not covered in this paper) can be applied to assure that defects and vulnerabilities have not been introduced by any such modifications.

The online Smart Checklist provides guidance to process performers during the execution of the process. It gives information about the history of the current execution; the current state of the process, including what each of the teams is doing; and the options that can be taken going forward from the current state. It is intended to improve situational awareness and provide detailed guidance in both complex and unusual situations. Since it has been repeatedly shown that exceptional situations significantly increase the incidence of errors, our approach has tried to explicitly support recognizing such situations and representing how to handle them if they do arise. Finally, after completion of the procedure, the Smart Checklist System can generate a detailed description of the process execution that just occurred. This documentation can be further annotated by the clinicians and included in the patient record. It also provides an important source of data for evaluating low-level procedural differences and their impact on outcomes.

To evaluate our approach, we have presented walk-throughs and mock-ups to focus groups of medical professionals and to human interface design experts. The approach has been well received, with medical professionals recognizing its potential to reduce errors, improve documentation and training. The medical professionals confirmed that it would be useful to show past process steps in addition to current pending steps. They thought that being able to access the process execution history would be particularly helpful during hand-offs. They noted that the step hierarchy shown on the Smart Checklist would be useful as a way to visualize the structure of the process. Some of them mentioned that this hierarchy would be particularly helpful when browsing through the process execution history, as this history could be potentially long, but the hierarchy may not be necessary for the current steps that are in progress, as usually there will be a small number of these. Earlier focus groups confirmed our expectations that in some situations it would be useful to access the tasks of other performers as well to get a better sense of the overall execution state of the process and also suggested that it might be useful if the Smart Checklist facilitated communication between different process performers working together as a team. In our most recent review, the focus group members were pleased with the team view representation presented in this paper, finding it easy to understand their own team's view and its pending interactions with other teams. Their feedback led us to provide better support for displaying shared data.

Suggestions that we have yet to act on include providing different user modes, UNDO capabilities, and better inter-team communication support. User modes would allow novices, for example, to easily request more guidance without burdening experts with unnecessary explanations. An UNDO option would allow an erroneous entry to be deleted, but this has to be done with care so that an actual audit trail cannot be inappropriately modified. Inter-team communication would need to be integrated with a hospital's communication systems. The focus groups expressed strong approval for the emphasis and support that is provided for recognizing and dealing with exceptional situations and suggested some ideas for additional support. For example, they pointed out that a detailed list of possible problems that might arise during the performance of an activity would be useful even for expert process performers. Such a list could be derived from the process model and made part of the generated Narration. Based on their feedback, we are currently trying to incorporate some of the standardized crisis checklists (e.g., [15]) that have been developed into our Smart Checklist system. For example, in our **separate from bypass** subprocess model, the step **follow protamine reaction protocol** is based on a customized crisis checklist developed at the VABHS [16].

In the near future, we plan to conduct experiments to evaluate the Smart Checklists in simulated surgeries, where the impact of the Smart Checklists on the incidence of errors can be evaluated. We plan to focus on common, high-stakes episodes in cardiac surgery, especially ones in which complex exceptional situations arise. These include phases of surgery such as heparinization, weaning from the cardiopulmonary bypass pump, and protamine administration. We are also recording measures of the cognitive load on members of the cardiac surgery team during procedures. It is well-established that errors are more likely when process performers are under high cognitive load [1]. While the Smart Checklist is expected to help reduce errors by explicitly reminding team members of the steps that need to be performed in particular contexts and more generally by improving their situational awareness, especially after interruptions and distractions, it could also be used to directly reduce load by, e.g., signaling that a team member should not be interrupted by non-critical communication at a particular time. Eventually, we hope to be able to use discrete event simulations of the process model to detect impending high cognitive load situations.

Finally, we are considering how our work could complement work in the new area of Surgical Data Science [17]. We expect that the progress being made in that area will yield events, insights, and measurements that will be useful in further iterative improvements to our models. Conversely, we believe that our work should be of substantial benefit to the Surgical Data Science community by suggesting hierarchical frameworks within which to structure observed and recorded data streams. As noted above, context plays a very important role in understanding and supporting surgical processes. While it may be possible to create such contexts by inferring hierarchy and other forms of structure from raw surgical event streams, we believe that, in making such structure explicit, our work will provide a framework for creating useful contextual information from the surgical data event streams that are collected.

Acknowledgments

The authors thank the following VABHS staff, Shosha Beal, Jacquelyn Quin, Miguel Haime, Rithy Srey, Geoffrey Rance, and Suzana Zorca for their help with the process modeling efforts, and Jennifer Gabany for helping make the whole project happen. The authors also thank Julian Goldman and David Arney for their help applying OpenICE,

Elizabeth Henneman and Jenna Marquard for their input on the Single-Team Smart Checklist, and Krisandra Cusanelli for her work on the Multi-Team Smart Checklist design. This material is based on work supported by the National Institutes of Health (NIH) under Award 1R01HL126896. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the NIH.

References

1. J. A. Wahr, R. L. Prager, J. H. Abernathy III, E. A. Martinez, et al. Patient safety in the cardiac operating room: Human factors and teamwork: A scientific statement from the American Heart Association. *Circulation*, 128(10):1139–69, 2013.

2. A. G. Cass, B. S. Lerner, E. K. McCall, L. J. Osterweil, J. S. M. Sutton, and A. Wise. Little-JIL/Juliette: A process definition language and interpreter. In *Proc 22nd Intl Conf Softw Eng*, pages 754–757, Limerick, Ireland, 2000.

3. G. S. Avrunin, L. A. Clarke, L. J. Osterweil, et al. Experience modeling and analyzing medical processes: UMass/Baystate medical safety project overview. In *1st ACM Intl Health Inform Symp*, pages 316–325, Arlington, VA, Nov. 2010.

4. W. C. Mertens, S. C. Christov, G. S. Avrunin, et al. Using process elicitation and validation to understand and improve chemotherapy ordering and delivery. *Jt Comm J Qual Patient Saf*, 38(11):497–505, Nov. 2012.

5. D. A. Wiegmann, A. W. ElBardissi, and J. A. Dearani. Disruptions in surgical flow and their relationship to surgical errors: an exploratory investigation. *Surgery*, 142(5):658–65, Nov. 2007.

6. ASTM International. *ASTM F2761-2009. Medical Devices and Medical Systems—Essential Safety Requirements for Equipment Comprising the Patient-Centric Integrated Clinical Environment (ICE), Part 1: General Requirements and Conceptual Model*. 2009.

7. M. Peleg, S. W. Tu, J. Bury, et al. Comparing computer-interpretable guideline models: A case-study approach. *J Am Med Inform Assoc*, 10:2003, 2002.

8. OMG. Business process modeling notation (BPMN), version 2.0.1. http://www.omg.org/spec/BPMN/2.0.1/.

9. S. C. Christov, J. L. Marquard, G. S. Avrunin, and L. A. Clarke. Assessing the effectiveness of five process elicitation methods: A case study of chemotherapy treatment plan review. *J Appl Ergonomics*, 59:364–376, 2017.

10. T. S. De Silva, D. MacDonald, G. Paterson, K. C. Sikdar, and B. Cochrane. Systematized nomenclature of medicine clinical terms (SNOMED CT) to represent computed tomography procedures. *Comput Methods Prog Biomed*, 101(3):324–329, Mar. 2011.

11. B. M. Hales and P. J. Pronovost. The checklist: a tool for error management and performance improvement. *J Crit Care*, 21:231–235, 2006.

12. N. B. Moe and T. Dybå. The use of an electronic process guide in a medium-sized software development company. *Softw Process: Improvement and Practice*, 11(1):21–34, 2006.

13. S. Nan, P. Van Gorp, X. Lu, et al. A meta-model for computer executable dynamic clinical safety checklists. *BMC Med Inform Decis Mak*, 17(1):170, Dec 2017.

14. F. Lalys and P. Jannin. Surgical process modelling: A review. *Intl J Comput Assist Radiol Surg*, 9:495–511, 2014.

15. J. E. Ziewacz, A. F. Arriaga, A. M. Bader, et al. Crisis checklists for the operating room: Development and pilot testing. *J Am Coll Surg*, 213(2):212 – 217.e10, 2011.

16. S. A. Hirji, C. Tarola, H. Amirfarzan, et al. Utility and feasibility of intra- and postoperative crisis management checklists in cardiac surgery. In *Proc 14th Annual Society of Thoracic Surgeons Crit Care Conf*, Oct. 2017.

17. L. Maier-Hein, S. S. Vedula, S. Speidel, et al. Surgical data science for next-generation interventions. *Nat Biomed Eng*, 1:691–696, Sept. 2017.