A Formal Approach For Modelling And Analysing Surgical Procedures

Ioana Sandu[1 [0009−0005−4512−7325]], Rita Borgo[1 [0000−0003−2875−6793]], Prokar Dasgupta[2 [0000−0001−8690−0445]], Ramesh Thurairaja[3], and Luca Viganò[1 [0000−0001−9916−271X]]

[1] Department of Informatics, King's College London, UK
[2] Peter Gorer Department of Immunobiology, King's College London, UK
[3] Urology Department, Guy's & St Thomas' Hospital NHS Foundation Trust, UK
{ioana.sandu, rita.borgo, prokar.dasgupta, luca.vigano}@kcl.ac.uk, Ramesh.Thurairaja@gstt.nhs.uk

Abstract. Surgical procedures are often not "standardised" (i.e., defined in a unique and unambiguous way), but rather exist as implicit knowledge in the minds of the surgeon and the surgical team. This reliance extends to pre-surgery planning and effective communication during the procedure. We introduce a novel approach for the formal and automated analysis of surgical procedures, leveraging established techniques developed for the analysis of security ceremonies. Our approach allows us to model as mutations the variants of a procedure and the mistakes that members of the surgical team might make, and to automatically identify violations of the intended properties of a procedure.

Keywords: Formal Methods · Mistakes in Surgical Procedures · Variants of Surgical Procedures · Security Ceremonies.

1 Introduction

Context and Motivation. This paper is the result of a collaboration between computer scientists and clinician scientists, which commenced with the live observation of a robot-assisted prostatectomy and cystectomy, leading to in-depth discussions on the actual execution of surgical procedures. These emphasised that much of a surgical procedure is often in the heads of the surgeon and of the members of the surgical team. This reliance on internalised knowledge hinges on two critical activities: (1) comprehensive pre-surgery discussions between the team members, (2) effective communication throughout the procedure. Recognising the potential for errors in both activities, which could jeopardise patient safety, *surgical process models (SPMs)* have been proposed to represent surgical procedures. These models offer "simplified, formal, or semi-formal representations of a network of surgery-related activities" [7].[4] SPMs draw upon concepts from workflow management and computer science, and (often) provide a representation of a surgical procedure that can be communicated to team members

[4] See [3, 5, 14, 17] for SPMs for robot-assisted prostatectomies and similar procedures.

as well as to other surgeons so they may follow the same steps. However, even in the case of more formal SPMs, little to no attention has been devoted to using SPMs to reason about procedures, particularly in the context of (1) and (2).

Contributions. We propose a different approach but one still anchored in computer science and, more specifically, cybersecurity: we formally model and reason about surgical procedures by representing them as security ceremonies. Modelling a surgical procedure as a security ceremony brings some important advantages. It provides conceptual clarity and allows one to represent the procedure as a message sequence chart that can be published and shared with others. It also provides a structured framework for reasoning as it enables us to adapt to surgical procedures established methodologies and automated approaches developed for the formal analysis of security ceremonies and their properties.

A *security protocol* is essentially a communication protocol (an agreed sequence of actions performed by agents communicating to accomplish some mutually desirable goals) that makes use of cryptographic techniques, allowing the communicating agents to satisfy security properties, such as authentication, or confidentiality or integrity of data. A *security ceremony* expands a security protocol to include human nodes alongside computer nodes, with communication links that comprise user interfaces, human-to-computer and human-to-human communication, and transfers of physical objects that carry data [13]. Hence, a ceremony's analysis should include the mistakes that human agents might make when they execute their tasks. Modelling a surgical procedure as a security ceremony thus allows us to consider *mutations* of the ceremony/procedure that formalise possible mistakes made by members of the surgical team. For instance, we formally model that the actions of the surgical team should not cause internal bleeding or endanger the patient, and our approach allows us to capture violations of this property, e.g., situations where a surgeon performs an internal incision without the assistant applying clips to prevent bleeding.[5] To automatically identify that such mistakes violate the intended properties of the procedure, we adapt and extend the mutation-based analysis approach proposed in [9] (but we use a different tool, UPPAAL [4], as it provides greater visual simplicity). Also, mutations provide the means for researchers and surgeons to explore variants of the procedure (e.g., alterations in the order of actions) and check if they lead to property violations without having to perform the variant on a live patient.

To illustrate our approach, we consider two stages of a laparoscopic prostatectomy procedure that is described informally in [3], which provides one of the most comprehensive descriptions of a laparoscopic prostatectomy. However, the description in [3], as is standard in such papers, is purely textual (with a few anatomical illustrations) and thus informal and prone to misunderstandings, so providing a formal model as we do is already a valuable contribution.[6]

[5] We focus on mistakes by human agents, but mistakes by robotic agents could be considered similarly.

[6] Note that [3] is 20+ years old and some specifics of the standard laparoscopic prostatectomy procedure might have changed in the meantime (cf. the newer papers discussed in §2) but adapting our models and analysis accordingly would be quite easy.

Structure. In §2, we discuss background and related work. We present our formal model in §3, the mutations in §4, and the formal analysis in §5. We draw conclusions in §6. Full details of our models and analyses are provided in [8].

2 Background and Related Work

Robotic-assisted surgery (RAS) has transformed the conventional operating room by introducing changes that include increased spatial requirements due to equipment and the physical separation of console surgeons from patients and team. In contrast to traditional arrangements [10], the configuration of RAS may hinder interpersonal cues and lead to potential miscommunication. Our approach proposes to reason about surgical procedures by conceptualising them as security ceremonies, which offer an explicit representation of human agents and their communications with other agents (human or not) [2, 9]. This perspective enables us to systematically incorporate and reason about human mistakes in the context of RAS or surgical procedures of any kind. We could similarly model robotic agents in RAS and other features of such procedures. This is important as RAS encompasses the patient, surgery type, surgical goals, tasks contributing to those goals, patient-related factors, and situational factors. The integration of new technologies into the operating room has the potential to significantly alter the prerequisites for effective teamwork, procedural workflows, and individual skills [11]. The distinctive setup of RAS introduces new challenges in maintaining situational awareness, team coordination, and information exchange [15]. Hence, effective communication is crucial for maintaining a surgeon's situational awareness, especially when operating from the console [16]. Communication, a recognised source of disruption in surgeries, has been undergoing fundamental changes in RAS due to the relocation of the surgeon from the operating table, and the impact of workflow disruptions/interruptions is explored in, e.g., [15, 16]. Specific verbal/non-verbal cues are crucial for team coordination [1], and studies have delved into the influence of anticipation and teamwork in RAS [10].

Excellent methods for conventional laparoscopic radical prostatectomy have been described in, e.g., [3, 5, 17], but there is currently no standardised surgical technique for robot-assisted radical prostatectomy.

In this paper, we demonstrate the efficacy of our approach on a simple stage of the procedure, the cutting stage, and on a more intricate stage, the dissection of the lateral surfaces of the prostate. The latter stage is pivotal because preserving the neurovascular bundles is paramount for ensuring a successful surgical outcome for patients who aim to maintain postoperative potency. Failure to preserve these bundles could significantly impact such patients' recovery. Various approaches to nerve-sparing prostatectomies are discussed in [6]. Denonvilliers' fascia is a crucial structure covering the posterior surface of the prostate and separating it from the rectum. It plays a vital role in the confinement of cancer within the prostate and facilitating an operation without damaging the nerves responsible for erectile function and continence, while ensuring the removal of all neoplastic tissue [12]. Hence, this stage not only demonstrates the close col-

laboration between surgeon and assistant but also allows us to reason about one of the key factors contributing to a successful outcome and recovery.

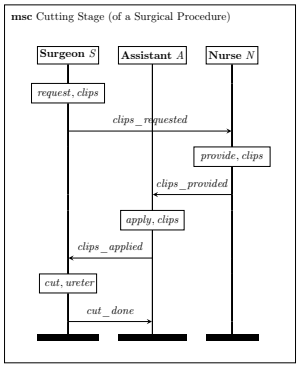
3 Formal Model

In a surgical procedure, multiple agents collaborate through a series of orchestrated actions and message exchanges to execute their tasks seamlessly as a cohesive team. We thus model a surgical procedure as a sequence of actions and messages exchanged so that other actions can occur. As a concrete example, we provide a formal model (and security analysis) of two stages of a laparoscopic prostatectomy procedure that is described informally in [3]. The *message sequence chart (MSC)* in Fig. 1a shows the cutting stage of this procedure, where three agents, a surgeon S, an assistant A and a nurse N, collaborate to carry out an internal incision on a patient.[7] Fig. 1c shows the MSC that we have drawn for the *lateral dissection stage* (the dissection of the lateral surfaces of the prostate), where VD, DF, SV and NVB abbreviate Denonvilliers' Fascia, vasa deferentes, seminal vesicles and neurovascular bundles, respectively.

In an MSC, each agent is defined as a process characterised by a series of surgical actions (the boxes in each agent's vertical timeline) and messages they send (the horizontal arrows), confirming that they have carried out an action or requested other agents to carry out an action. As is standard for security protocols/ceremonies (e.g., [9, 13]), we define the *algebra of messages* as $\mathcal{T}_{\Sigma}(\mathcal{V})$. The signature Σ contains possibly disjoint sets of constants (e.g., representing agent names and other publicly known values) and $\mathcal{V}$ is a countably infinite set of variables. Σ can easily be extended to include function symbols to formalise symmetric and asymmetric decryption and other cryptographic operators. Given the set M of all messages that can be built according to the algebra, we define for each agent Ag the sets M^s_{Ag} and M^r_{Ag} of messages Ag can send and receive. Here, we only consider messages that can be defined as constants, as that is what our case study requires. For instance, for S in both stages of the procedure we define $M^s_S = \{clips\_requested, cut\_done, VD\_and\_SV\_pulled, pedicle\_dissected, PFS\_entered, visceral\_fascia\_incised, DF\_incised\}$. Our approach can accommodate more complex messages, e.g., that contain random numbers and are encrypted. Moreover, we consider only honest agents who behave according to what the surgical procedure expects, but below we will extend this to consider mistakes by agents (and we could also consider dishonest agents who can do anything they want as is standard in formal analysis of security ceremonies).

In formal analysis of security ceremonies, and thus in our approach, agents are formalised as processes that represent the vertical lines in an MSC and that are often called role-scripts. A *role-script* is a sequence of events $e \in T_{\Sigma \cup RoleActions}$, where $RoleActions = \{Snd, Rcv, s\_action, Start\}$ is a set of action names with their respective arity. For example, the role-script of the cutting stage of S is shown in Fig. 1b. $Start(Ag,\ K^0_{Ag})$ is the first event of a role-script and it takes

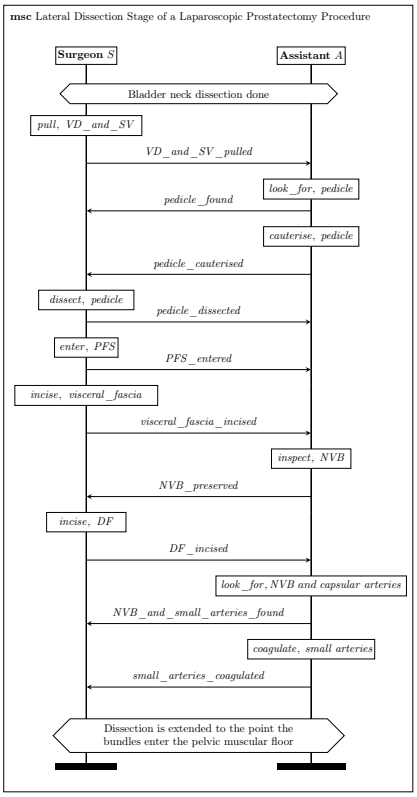
[7] Note that this cutting stage is quite general and could be applied also to other surgical procedures and not just to a prostatectomy.



(a) MSC of the Cutting Stage

$$
\begin{aligned}
&RoleScript_S = \\
&\quad [Start(S,\ K_S^0) \\
&\quad\ s\_action(S,\ request,\ clips) \\
&\quad\ Snd(S,\ N,\ clips\_requested) \\
&\quad\ Rcv(A,\ S,\ clips\_applied) \\
&\quad\ s\_action(S,\ cut,\ ureter) \\
&\quad\ Snd(S,\ A,\ cut\_done)]
\end{aligned}
$$

(b) Role-script of S



(c) MSC of the Lateral Dissection Stage

Fig. 1: MSCs of Two Stages of a Laparoscopic Prostatectomy Procedure and Role-Script of the Surgeon in the Cutting Stage

place once, where K_{Ag}^0 is the *initial knowledge* of agent Ag at the beginning of the process (typically, it contains the names of the other agents and the constant messages Ag will send). Ag's *knowledge* increases monotonically as Ag receives messages. *Snd* and *Rcv* events are of the form $Snd(Ag_s,\ Ag_r,\ m)$ and $Rcv(Ag_s,\ Ag_r,\ m)$, where Ag_s is the sender of the message, Ag_r is the receiver and m is the message that is being sent or received. In our model, the messages have only one recipient to indicate who the information concerns; this is primarily a modeling choice for clarity, but it can be adjusted so that multiple agents receive the same message. Moreover, we focus on secure communication channels between agents, but our approach extends to various types of channels (e.g., authenticated or insecure ones, which can be attacked by a dishonest agent).

Surgical action events represent the actions each agent performs to ensure the progression of the surgery and are defined as $s\_action(Ag,\ a_{Ag}, O)$, agent Ag performs action a_{Ag} on or with object O. For the two stages we are considering, $a_S \in \{request,\ cut,\ pull,\ dissect,\ enter,\ incise\}$, $a_A \in \{apply,\ look\_for,$

cauterise, *inspect*, *coagulate*} and $a_N \in \{provide\}$.[8] The objects are $O = \{clips,\ ureter,\ VD\_and\_SV,\ pedicle,\ PFS,\ visceral\_fascia,\ NVB,\ DF,\ capsular\ arteries,\ small\ arteries\}$, which are significant as the same action may be performed multiple times for different objects (e.g., one can request clips or a scalpel), or an action could be performed with the same object but at different stages of the procedure (e.g., clips can be requested during cutting or suturing).

Our approach is based on an *execution model* that is defined by a *multi-set rewriting system* like in many security protocol/ceremony analysis approaches and tools. A *state* is a multiset of facts that model resources, including the information that agents know and exchange. Formally, the state $S^i_{Ag} = \{i;\ K^i_{Ag}\}$ of agent Ag is characterised by the state number i and the knowledge $K^i_{Ag} = K^0_{Ag} \cup \{messages\ received\ by\ Ag\}$ that Ag possesses at i, and $S^i = \{i;\ \{K^i_{Ag} \mid Ag \text{ is an agent}\}\}$ represents the state of all agents at that point in the execution.

A *trace* is a finite sequence of multisets of role-actions and is generated by the application of *state transition rules* of the form

$$prem \xrightarrow{finite\ sequence\ of\ role-actions\ and\ internal\ checks} conc$$

which is applicable when the current state matches the premise *prem* and the internal checks on the messages received are satisfied. These checks are typically not displayed in a role-script but only act as guards. The rule's application produces the conclusion *conc* (a new state) and records the instantiations of role-actions in the trace. For instance, for the cutting stage,

$$\{1;\ K^1_S, K^1_A, K^1_N\} \xrightarrow{Rcv(S,\ N,\ ?X),\ Check(?X\ =\ clips\_requested),} \xrightarrow{s\_action(N,\ provide,\ clips),\ Snd(N,\ A,\ clips\_provided)} \{2;\ K^2_S, K^2_A, K^2_N\}$$

represents a transition from state S^1 to state S^2, where we split the arrow for readability and where we are numbering the states from 0 to 5 assuming that the cutting stage is the initial stage of the procedure (if it is not the initial stage, then the states will be numbered differently but still consecutively). Agent N receives a message $?X$ from S, checks the contents of that message, performs a surgical action and sends a confirmation message to A. We write that agent Ag receives $?X$ to allow Ag to check, via $Check(?X = m)$, that this is indeed the message m that Ag was expecting. This check will become useful later as it will enable us to consider mistakes that agents could make, such as changing the contents of the message or sending the wrong message.[9]

Nurse N can carry out only the transition above. Surgeon S's can carry out

$$\{0;\ K^0_S, K^0_A, K^0_N\} \xrightarrow{Start,\ s\_action(S,\ request,\ clips),\ Snd(S,\ N,\ clips\_requested)} \{1;\ K^1_S, K^1_A, K^1_N\}$$

[8] We could represent cuts and incisions by means of a single action but we prefer to consider two distinct actions *cut* and *incise* to distinguish between actions that might use different instruments (e.g., scissors or scalpels).

[9] In security ceremonies, there typically is a ? also in front of the sender's name in a *Rcv* event. This allows one to consider an attacker that is claiming to be the sender. Here, we avoid doing so given that we are not yet explicitly considering an attacker.

$$\{3;\ K_S^3, K_A^3, K_N^3\} \xrightarrow{Rcv(A,\ S,\ ?X),\ Check(?X\ =\ clips\_applied),} \xrightarrow{s\_action(S,\ cut,\ ureter),\ Snd(S,\ A,\ cut\_done)} \{4;\ K_S^4, K_A^4, K_N^4\}$$

and assistant A can carry out

$$\{2;\ K_S^2, K_A^2, K_N^2\} \xrightarrow{Rcv(N,\ A,\ ?X),\ Check(?X\ =\ clips\_provided),} \xrightarrow{s\_action(A,\ apply,\ clips),\ Snd(A,\ S,\ clips\_applied)} \{3;\ K_S^3, K_A^3, K_N^3\}$$

$$\{4;\ \ K_S^4, K_A^4, K_N^4\} \xrightarrow{Rcv(S,\ A,\ ?X),\ Check(?X\ =\ cut\_done)} \{5;\ \ K_S^5, K_A^5, K_N^5\}$$

The expected sequence of actions is then given by the rules applied in the same order as their state numbers, mirroring the MSC. It captures the unaltered process, when all agents execute their tasks precisely as expected. No errors are occurring, but rather every action unfolds in a seamless sequence, with agents patiently awaiting messages from their predecessors before proceeding.

The transition rules for the lateral dissection stage are similar, e.g.

$$\{i;\ K_S^i, K_A^i, K_N^i\} \xrightarrow{Rcv(A,\ S,\ ?X),\ Check(?X\ =\ NVB\_preserved),} \xrightarrow{s\_action(S,\ incise,\ DF),\ Snd(S,\ A,\ DF\_incised)} \{i+1;\ K_S^{i+1}, K_A^{i+1}, K_N^{i+1}\}$$

represents the incision of the DF between states i and $i + 1$ (we use i to indicate the state as it occurs in the full procedure when all stages are considered).

4 Mutations

When engaged in a surgical procedure (and, in general, in a security ceremony), humans might make mistakes because of various reasons, such as communication errors, distraction, inexperience, stress, etc. These mistakes alter the process flow and create deviations of the original ceremony specification that may impact the security of the ceremony or, in a surgical procedure, the safety of the patient.

We adapt and extend to surgical procedures the approach of [9], which allows security analysts to model mistakes by human agents as *mutations* of the behaviour that the ceremony originally specified for such agents.[10] Mutations thus create alternative formal specifications of the original ceremony, which we can then formally analyse to see if they lead to violations of the intended properties (and thus endanger patient safety). Studying these mutations is also interesting as they might reveal alternative ways to carry out the procedure that do not endanger the patient and are, possibly, faster or more efficient. In this paper, we

[10] These mutations refer to deviations from the expected sequence of actions in a process or procedure, not to the mutations found in other fields such as molecular biology or genetic mutations. We use the mutation names given in [9] but other names have been proposed for similar mutations in different disciplines, e.g., in biology.

focus on formal analysis, dis-/proving properties of a procedure, but in the future we plan to also carry out a cost analysis of the different secure alternatives.

Since mutations allow humans to do things that were not foreseen in the original procedure, we formalise them by introducing new transition rules that are themselves mutations of the original ones. For surgical procedures, we focus on two mutations, skip and replace (but more could be considered). The *skip mutation* enables us to formalise an agent skipping some actions that the surgical procedure expects them to carry out. For instance, for the cutting stage, the case in which A does not apply the clips but nonetheless sends a message confirming task completion is formalised by the mutated rule

$$\{2;\ \ K_S^2, K_A^2, K_N^2\} \xrightarrow{Rcv(N,\ A,\ ?X),\ Check(?X\ =\ clips\_provided),} \xrightarrow{\sout{s\_action(A,\ apply,\ clips)},\ Snd(A,\ S,\ clips\_applied)} \{3;\ \ K_S^3, K_A^3, K_N^3\}$$

and the case in which A applies the clips but does not send a confirmation message is formalised by the mutated rule

$$\{2;\ \ K_S^2, K_A^2, K_N^2\} \xrightarrow{Rcv(N,\ A,\ ?X),\ Check(?X\ =\ clips\_provided),} \xrightarrow{s\_action(A,\ apply,\ clips),\ \sout{Snd(A,\ S,\ clips\_applied)}} \{3;\ \ K_S^3, K_A^3, K_N^3\}$$

In the *replace mutation*, an agent performs their actions as expected but replaces a message with another one. For instance, in case of complex messages consisting of different components, an agent could send just part of the message by mistake, as considered in the mutations in [9]. For our case study, where messages are simple, we introduce the novel (w.r.t. [9]) concept of *negative message*, which we write as "not_m", e.g., $not\_clips\_applied$, and we extend accordingly the sets of messages agents can send or receive. This allows agents in the cutting stage of our case study to execute mutated rules such as:

$$\{2;\ \ K_S^2, K_A^2, K_N^2\} \xrightarrow{Rcv(N,\ A,\ ?X),\ Check(?X\ =\ clips\_provided),} \xrightarrow{s\_action(A,\ apply,\ clips),\ Snd(A,\ S,\ not\_clips\_applied)} \{3;\ \ K_S^3, K_A^3, K_N^3\}$$

Each action in a surgical procedure has a purpose and altering even a single action will cause some sort of propagation of the mistake for the next agents, which could impact patient safety. When a mutation happens, the other agents will not be able to carry out their actions unless their rules are mutated as well. For instance, if A does not apply the clips, then S will not cut and execution will deadlock. Our aim is for a procedure not to deadlock during execution but rather to be executed completely so that we can check whether the intended properties are satisfied even in presence of a mistake. To ensure that we only have executable traces, every mutation is matched via a *matching mutation* and propagated through a trace. A matching mutation for a skip mutation depends on the ability of an agent to perform their action given that the previous agent has skipped theirs. For example, if N skips their action to provide the clips

$$\{1;\ \ K_S^1, K_A^1, K_N^1\} \xrightarrow{Rcv(S,\ N,\ ?X),\ Check(?X = clips\_requested),} \xrightarrow{\sout{s\_action(N,\ provide,\ clips)},\ Snd(N,\ A,\ clips\_provided)} \{2;\ \ K_S^2, K_A^2, K_N^2\}$$

the matching mutation formalises that the action A was going to perform, apply clips, is skipped as it is impossible for A to apply clips unless N provides them

$$\{2;\ \ K_S^2, K_A^2, K_N^2\} \xrightarrow{Rcv(N,\ A,\ ?X),\ Check(?X = clips\_provided),} \xrightarrow{\sout{s\_action(A,\ apply,\ clips)},\ Snd(A,\ S,\ clips\_applied)} \{3;\ \ K_S^3, K_A^3, K_N^3\}$$

Other mutations are matched and propagated similarly (cf. [8] for details).

5 Formal and Automated Analysis

Surgical procedures should first and foremost guarantee patient safety. Hence, everything that might endanger it should be avoided and formally specified as a *property* to be satisfied. This way, we can formally analyse it and either prove it to hold or, if not, produce a trace that shows the sequence of actions violating the property (if the tool terminates). We formalise properties using a linear temporal logic, which allows us to specify that if an event occurs now, then certain other events must have occurred in the past. For instance, the patient should not bleed out due to a negligent incision, i.e., we require that in all traces, if S carries out a cut at some time instant, then there must exist previous time instants, ordered temporally, in which the clips have been requested, provided and applied:

Property 1 (Clip-before-cutting). For all traces,

$$\begin{aligned} s\_action(S, cut)\ @l \implies\ & s\_action(S,\ request,\ clips)\ @i \\ & \&\ s\_action(N,\ provide,\ clips)\ @j \\ & \&\ s\_action(A,\ apply,\ clips)\ @k \\ & \&\ i < j < k < l \end{aligned}$$

Property 1, which is a general and quite obvious property of any surgical procedure but is also explicitly inspired by the informal discussion in [3], establishes a fundamental sequence of actions that occur in any procedure that includes a cutting stage (cf. Fig. 1a). The following three properties consider, instead, the lateral dissection stage (cf. Fig. 1c) and are again inspired by [3] as well as by the more recent [6, 12]. Property 2 pertains to the dissection of the pedicle, Property 3 checks whether the incision of DF has been performed, Property 4 expresses that preserving the NVB is crucial for potency recovery [6].

Property 2 (Dissection of the pedicle). For all traces,

$$\begin{aligned} s\_action(S, dissect, pedicle)\ @l \implies\ & s\_action(S,\ pull,\ VD\_and\_SV)\ @i \\ & \&\ s\_action(A,\ look\_for,\ pedicle)\ @j \\ & \&\ s\_action(A,\ cauterise,\ pedicle)\ @k \\ & \&\ i < j < k < l \end{aligned}$$

Property 3 (Incision of the Denonvilliers' Fascia). For all traces,

$$\begin{aligned} s\_action(S, incise, DF)\ @l \implies & s\_action(S, enter, PFS)\ @i \\ & \&\ s\_action(S, incise, visceral\_fascia)\ @j \\ & \&\ s\_action(A, inspect, NVB)\ @k \\ & \&\ i < j < k < l \end{aligned}$$

Property 4 (Check if the nerves are preserved). For all traces,

$$\begin{aligned} s\_action(A, inspect, NVB)\ @l \implies & s\_action(A,\ cauterise,\ pedicle)\ @i \\ & \&\ s\_action(S,\ enter,\ PFS)\ @j \\ & \&\ s\_action(S,\ incise,\ visceral\_fascia)\ @k \\ & \&\ i < j < k < l \end{aligned}$$

We automatically analysed these properties using UPPAAL [4]; see [8] for details on the modelling and analysis using UPPAAL and on the attack traces it outputs (and for other properties that could be considered). UPPAAL confirmed that Property 1 holds true across all traces when mutations are deactivated, but intriguing violations caused by agent mistakes become discernible upon enabling mutations (and matching mutations), e.g., the surgeon can execute a cut without the application of clips if mistakes occur due to miscommunication or negligence. Similarly, Property 2 suffers from an attack in which S dissects the pedicle without VD and SV being pulled and without the pedicle being cauterised. This result could indicate several possibilities: the pedicle might have been visible without the need to pull VD and SV and there might have been no bleeding requiring cauterisation. Although the patient's safety may not be compromised, this still represents a violation of the expected sequence of actions. Thus, it is crucial for a surgeon to interpret the results, as this attack could represent a legitimate shortcut in the procedure. Our approach thus allows clinicians to consider variants of the procedure and reason about them.

Property 3 is violated when S makes an incision of the DF without incising the visceral fascia and without A inspecting the NVB. Property 4 is violated when the NVB have been inspected without entering the pericapsular fatty space and without incising the visceral fascia. It is important to ensure that Property 4 holds because of the presence of numerous NVB fibers between the posterior and intermediate layers of the DF, which makes dissection in this area hazardous for erectile nerves and should be avoided in nerve-sparing procedures [12].

6 Concluding Remarks

We view this paper as the first step towards the full-fledged analysis of surgical procedures. We plan to encompass a complete laparoscopic prostatectomy (and other procedures) by modelling and analysing all stages holistically rather than independently. We will explore methodologies for performing a prostatectomy, involving an expanded set of agents, both human and robotic, capable of executing a broader range of actions and transmitting additional messages. We

will also consider telesurgeries, emphasising the importance of cryptography for secure communication between agents in the presence of possible attackers.

We also plan to consider other mutations, such as a mutation that "negates" an action by undoing it, which would, e.g., capture the mistake that occurs when A applies the clips but then removes them before S cuts, resulting in bleeding.

Although we could not discuss them here, our approach also allows one to explore variants of a procedure not only to identify attacks but also to study, say, an order of the actions different from that in the minds of the surgical team. Our objective is to offer the most suitable surgical approach for each individual, tailoring the procedure to meet specific patient needs and conditions.

References

1. Catchpole, K., et al.: Human factors in robotic assisted surgery: Lessons from studies 'in the Wild'. Applied ergonomics **78**, 270–276 (2019)
2. Curzon, P., Rukšėnas, R.: Modelling the user. In: The Handbook of Formal Methods in Human-Computer Interaction, pp. 211–245. Springer (2017)
3. Guillonneau, B., Vallancien, G.: Laparoscopic Radical Prostatectomy: The Montsouris Technique. The Journal of Urology **163**(6), 1643–1649 (2000)
4. Kim G. Larsen, Paul Pettersson, P.W.Y.: Uppaal (1995), https://uppaal.org
5. Martini, A., et al.: Contemporary techniques of prostate dissection for robot-assisted prostatectomy. European Urology **78**(4), 583–591 (2020)
6. Moschovas, M.C., Patel, V.: Neurovascular bundle preservation in robotic-assisted radical prostatectomy: How I do it after 15.000 cases. Int Braz J Urol **48** (2022)
7. Neumuth, T.: Surgical process modeling. Innov. Surg. Sci. **2**(3), 123–137 (2017)
8. Sandu, I., Borgo, R., Dasgupta, P., Thurairaja, R., Viganò, L.: A Formal Approach For Modelling And Analysing Surgical Procedures (Extended Version) (2024), https://arxiv.org/abs/2408.05001
9. Sempreboni, D., Viganò, L.: A Mutation-Based Approach for the Formal and Automated Analysis of Security Ceremonies. J. Comput. Secur **31**(4), 293–364 (2023)
10. Sexton, K., et al.: Anticipation, teamwork and cognitive load: chasing efficiency during robot-assisted surgery. BMJ quality & safety **27**(2), 148–154 (2018)
11. Souders, C.P., et al.: Flow disruptions in robotic-assisted abdominal sacrocolpopexy: does robotic surgery introduce unforeseen challenges for gynecologic surgeons? Int. Urogynecol. J. **30**(12), 2177–2182 (Dec 2019)
12. Tzelves, L., Protogerou, V., Varkarakis, I.: Denonvilliers' fascia: The prostate border to the outside world. Cancers (Basel) **14**(3) (Jan 2022)
13. Viganò, L.: Formal Methods for Socio-technical Security (Formal and Automated Analysis of Security Ceremonies). In: Coordination. LNCS 13271, Springer (2022)
14. Villers, A., et al.: Robot-assisted partial prostatectomy for anterior prostate cancer: a step-by-step guide. BJU International **119**(6), 968–974 (2017)
15. Weber, J., et al.: Effects of flow disruptions on mental workload and surgical performance in robotic-assisted surgery. World journal of surgery **42**, 3599–3607 (2018)
16. Weigl, M., et al.: Associations of intraoperative flow disruptions and operating room teamwork during robotic-assisted radical prostatectomy. Urology **114** (2018)
17. Zhou, X., et al.: Transvesical robot-assisted radical prostatectomy: initial experience and surgical outcomes. BJU international **126**(2), 300–308 (2020)