Combining YAWL and DBNs for Surgical Phase Detection

Patrick Philipp

Vision and Fusion Laboratory
Institute for Anthropomatics
Karlsruhe Institute of Technology (KIT), Germany
p.philipp@kit.edu

Technical Report IES-2016-01

Abstract: To provide assistance functions in context of surgical interventions, the use of a surgical phase detection plays an important role. By assessing the progress of an on-going surgery, a tailored (i.e., context sensitive) decision support for medical practitioners can be enabled. Subsequently, this provides opportunities to prevent errors, injuries, negligence or malpractices. In this work, a surgical phase detection, combining Yet Another Workflow Language (YAWL) with Dynamic Bayesian Networks (DBNs) is proposed. Thereby, YAWL is used to model the relationship of surgical phases; DBNs are used to allow for the detection of surgical phases of interest. The approach is presented for the application example of a cholecystectomy (removal of the gallbladder).

1 Introduction

In modern medicine, the use of assistance functions becomes increasingly important [PFHB16]. Such functions can be realized as part of a computer assisted surgery (CAS) to enable a decision support of the medical practitioners [KWN+15]. Thereby, a decision support opens up a scope of optimization: E.g. concerning the prevention of errors, injuries, negligence or (subsequently) malpractices.

In this context, a surgical phase detection plays an important role. Namely, because by assessing the progress of an on-going surgery, a tailored (i.e., context sensitive) decision support during an intervention can be enabled. In doing so, there is not only a passive dissemination (e.g. distribution via print media) of support (e.g. medical guidelines) – which was shown has only little effect on the actual practitioners behavior [FL92, SGM$^+$11]. Instead, we propose to provide an interactive assistance in terms of a context sensitive decision support to assist medical practitioners during an intervention.

In this work, we focus on the application example of the removal of the gallbladder (cholecystectomy). Thereby, we consider the standard procedure of a cholecystectomy, the laparoscopic cholecystectomy. It is a minimal invasive approach using laparoscopes. These are endoscopes, specialized for an abdominal surgery (i.e. a surgery concerning the stomach). It is therefore not surprising that regarding CAS, the considered procedure can be categorized as a computer-assisted abdominal surgery [KWN$^+$15].

This contribution is structured as follows: first, in sections 2 and 3, the fundamentals of Yet Another Workflow Language (YAWL) and Dynamic Bayesian Networks (DBNs) are presented. Section 4 gives details on the application example of a laparoscopic cholecystectomy and the according model in YAWL. Section 5 focuses on the modelling approach of combining YAWL and DBNS. The approach is verified in section 6 and, finally, a conclusion is drawn in section 7 on page 12.

2 Fundamentals on YAWL

There are various reasons for choosing YAWL (Yet Another Workflow Language) as a modeling tool for workflows [HRAA10]. One aspect is the expressive power reflected in various so called workflow patterns. Another aspect is the formal semantics which enable the application of analysis tools like WofYAWL [VvdAtH06] to verify a model.

Nevertheless, YAWL is said to be comprehensive [HRAA10], and building a valid model can be challenging task, especially if the medical expert uses YAWL on his own – and therefore without the assistance of a technical domain expert. Consequently, in [Phi16] we introduced translation rules transferring one comprehensible UML activity into a YAWL specification. That means, YAWL can be used as an intermediate language for further assistance functions which are build upon a YAWL specification.

Figure 2.1 depicts important elements of the YAWL notation. Since YAWL is an extension of so-called Workflow Nets [AH03], their elements show a high degree of similarity. Formally, a YAWL specification is a non-empty set of extended workflow nets (EWF Nets). Such an EWF Net can comprise several conditions

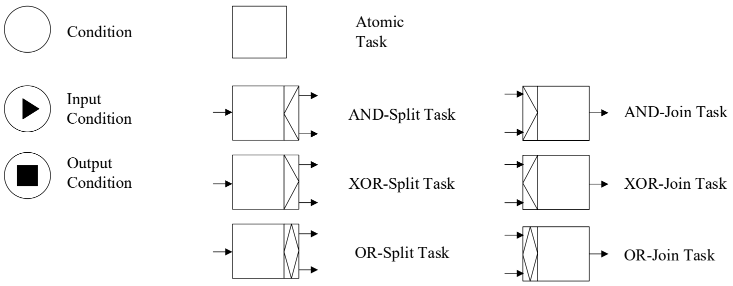

Figure 2.1: Important elements of the YAWL notation. A condition is represented by a circle and a task is represented by a square. There are two unique conditions: the so called input condition (black triangle in a circle) and the output condition (black square in a circle). A task can be expanded by a split and a join behavior which is indicated by the corresponding symbols.

which are represented by circles[1] (cf. Figure 2.1). Furthermore, an EWF Net comprises two unique conditions: one unique input condition, which is represented by a black triangle in a circle, as well as a unique output condition, which is represented by a black square in a circle (cf. Figure 2.1). A task of the EWF Net is given by a square, which can be expanded by a split and a join behavior (cf. Figure 2.1).

Formally, an EWF Net is given by the following tuple [VvdAtH06, HRAA10]:

$$EWF = (i, o, C, T, F, f_{\text{split}}, f_{\text{join}}, f_{\text{rem}}, f_{\text{nofi}}) ,$$

where

- $i \in C$ is the input condition,
- $o \in C$ is the output condition,
- C is a set of conditions,
- T is a set of tasks,
- $F \subseteq ((C \setminus \{o\}) \times T) \cup (T \times (C \setminus \{i\})) \cup (T \times T)$ is the flow relation,
- Every node in the graph $(C \cup T, F)$ is on a directed path from i to o,

[1]Compared to Petri Nets, a condition can be interpreted as a place [AH03].

- $f_{\text{split}}: \ T \rightsquigarrow \{\text{AND, OR, XOR}\}$ [2] specifies the split behavior of each task,
- $f_{\text{join}}: \ T \rightsquigarrow \{\text{AND, OR, XOR}\}$ specifies the join behavior of each task,
- $f_{\text{rem}} : T \rightsquigarrow \mathbb{P}^+ (T \cup C \setminus \{i, o\})$ [3] specifies the tokens to be removed by emptying a part of the net,
- $f_{\text{nofi}} : T \rightsquigarrow \mathbb{N} \times \mathbb{N}^{\text{inf}} \times \mathbb{N}^{\text{inf}} \times \{\text{dynamic, static}\}$ specifies the multiplicity of each task.

In this work, we focus on the combination of a given YAWL specification with Dynamic Bayesian Networks (DBNs). Therefore we introduced the necessary basics of YAWL in this section and proceed with the fundamentals on DBNs in the following section.

3 Fundamentals on DBNs

There are many reasons for considering Dynamic Bayesian Networks (DBNs) [DK89] as a modeling tool for dynamic systems [Mur02]. With respect to the modeling of medical workflows, we opted for DBNs because they combine a reasonable tradeoff between expressiveness and complexity, and include probabilistic models that have proved to be successful in practice (e.g. Hidden Markov Models) [LJ14]. Additionally, due to their factorized state space, DBN models allow improved modularity and interpretability. In contrast to a Hidden Markov Model, their state space can be expressed in a factored form and not only as a single discrete random variable. Furthermore, concerning Kalman Filter Models, DBNs allow for arbitrary probability distributions (not only for unimodal linear-Gaussians) [Pad10].

A Bayesian Network (BN) is a probabilistic graphical model (PGM), combining graph theoretic approaches with approaches of probability theory. Consequently, a BN over random variables $X^{0:N} := X^0, \ldots, X^N$ is given by a pair

$$B = (G, P) \,.$$

Whereby G corresponds to a directed, acyclic graph (DAG)

$$G = (V, E) \,, \ \text{and} \tag{3.1}$$

[2]Please note: $\rightsquigarrow$ denotes a partial function. I.e., a task can have no specified split behavior, too.
[3]Please note: $\mathbb{P}^+(X)$ denotes the power set of X without the empty set: $\mathbb{P}^+(X) = \mathbb{P}(X) \setminus \{\emptyset\}$

$$P(X^{0:N}) = \prod_{n=0}^{N} P(X^n|\text{Pa}(X^n))\,, \tag{3.2}$$

corresponds to a joint probability distribution [KF09]. DAG G in (3.1) is used to define dependencies between random variables $X^{0:N}$. It is also known as the structure of the BN. The vertex set V represents the set of random variables, while a directed edge $V_i \to V_j$ of the set of edges E represents a direct dependency between two variables. A missing edge symbolizes the independence of these two variables (cf. Figure 3.1).

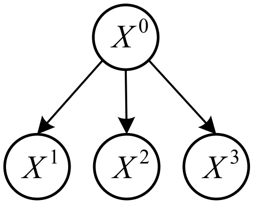

Figure 3.1: Graph of a Bayesian Network (BN) over random variables $X^{0:3}$. In this structure (also called Naive BN), there are directed edges connecting a root node (X^0) and its children ($X^{1:3}$). Since a missing edge symbolizes an independence of two variables, the joint probability $P(X^{0:3})$ can be factorized as follows: $P(X^{0:3}) = P(X^0)P(X^1|X^0)P(X^2|X^0)P(X^3|X^0)$.

The joint probability distribution in (3.2) is given by the product of all conditional probability distributions associated with the vertices of G. It is also known as the parameters of the BN. Here, Pa(X^n) denotes the set of parents of a random variable X^n. Graphically, this corresponds to vertices having a directed edge pointing to X^n's vertex. Please note, if Pa$(X^n) = \emptyset$, a random variable X^n is a root node of the BN, and $P(X^n|\,\emptyset) = P(X^n)$ gives the a-priori probability (cf. Figure 3.1).

A Dynamic Bayesian Network (DBN) is an extension of a BN, also taking the temporal dependencies of variables into account [Mur02]. A DBN is given by a pair

$$DBN = (B_0, B_{\to})\,,$$

where the BN B_0 uses $P(X_0^{0:N})$ to specify the a-priori probability distribution over random variables $X^{0:N}$ in a time step with index 0 (cf. Figure 3.1 as an example for a possible underlying DAG).

Furthermore, $B_{\to}$ specifies the conditional probability distribution over discrete

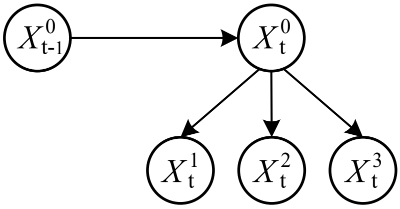

Figure 3.2: Simplified graph of a 2-Slice Temporal Bayesian Network (2TBN). In context of a DBN, a 2TBN (or: $B_{\rightarrow}$) is used as template for consecutive time steps t. For simplification, we omitted $X^{1:3}_{t-1}$, since in this example, there is no direct dependency to $X^{0:3}_t$.

time steps t by using

$$P(X^{0:N}_t | X^{0:N}_{t-1}) = \prod_{n=0}^{N} P(X^n_t | \text{Pa}(X^n_t)) \,. \tag{3.3}$$

In Equation (3.3), Pa(X^n_t) denotes the set of X^n_t's parents in the corresponding graph. The parents can be in the same time slice (e.g. representing instantaneous causation) or the previous one (i.e., we assume the model to be fist-order Markov). In the latter case, arcs point to time slices with ascending index, reflecting the causal flow of time [Mur02]. Please compare Figure 3.2 for an exemplary graph of $B_{\rightarrow}$ which is also known as a two-slice Temporal Bayesian Network (2TBN).

For T time-slices, the joint distribution of the DBN is given by equation [Mur02]:

$$\begin{aligned} P(X^{0:N}_{0:T}) &= \left(\prod_{n=0}^{N} P(X^n_0 | \text{Pa}(X^n_0)) \right) \prod_{t=1}^{T} \prod_{n=0}^{N} P(X^n_t | \text{Pa}(X^n_t)) \\ &= \prod_{t=0}^{T} \prod_{n=0}^{N} P(X^n_t | \text{Pa}(X^n_t)) \,. \end{aligned}$$

Graphically, this corresponds to an unrolling of the DBN, using B_0 as the initial distribution, and $B_{\rightarrow}$ as template for each following time slice. Refer to Figure 3.3 for a DBN unrolled for three time slices. Similar to HMMs, parameters of

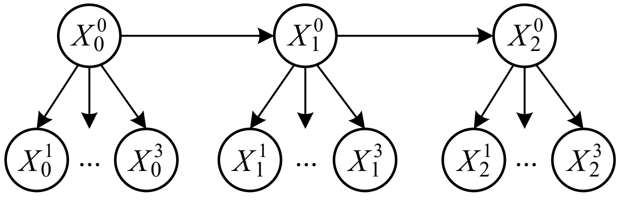

Figure 3.3: Example of a DBN unrolled for 3 slices using graphs depicted in Figure 3.1 and 3.2 as B_0 and $B_{\rightarrow}$. For simplification some nodes are omitted, which is graphically represented by three dots.

such a DBN, having N children, can be grouped as follows (cf. Figure 3.3):

$$P(X_0^0 = i) = \boldsymbol{\pi}(i) \tag{3.4}$$

$$P(X_t^0 = j | X_{t-1}^0 = i) = \boldsymbol{A}(i,j) \tag{3.5}$$

$$P(X_t^1 = j | X_t^0 = i) = \boldsymbol{B}^{(1)}(i,j) \tag{3.6}$$

$$\cdots$$

$$P(X_t^N = j | X_t^0 = i) = \boldsymbol{B}^{(N)}(i,j) \tag{3.7}$$

Equation (3.4) shows the inital probability distribution associated with the root node X^0 at time step $t = 0$ (X_0^0). Consequently, $\boldsymbol{\pi}(i)$ is a vector representing the a-priori probability of each of the values of X_0^0 being present. Please note that the statement $P(X = x)$ is a shorthand for an event $\omega \in \Omega : f_X(\omega) = x$, whereby the set of possible outcomes is denoted by Ω, and f_X maps an event ω to a possible value of a random variable X. We denote possible values x of X by $Val(X)$ [KF09].

In Equation (3.5) the probability distributions of the state transitions are given. With this, the dependencies over time (and between states) are expressed. Consequently, $\boldsymbol{A}(i,j)$ is an adjacency matrix extended by transition probabilities for entries unequal to 0. In Equations (3.6) and (3.7), the probability distribution for observations concerning the children of the root node are given. The naming of the matrices $\boldsymbol{A}(i,j)$ and $\boldsymbol{B}(i,j)$ is inspired by Hidden Markov Models (HMMs). Please note, that in case of HMMs the observation probabilities can be specified by a single matrix $\boldsymbol{B}(i,j)$, since the corresponding probability distribution cannot be factorized. That means, graphically, the root node would have only one child which incorporates the complete probability distribution.

4 Workflow of a Cholecystectomy

In Figure 4.1, the application example of this work, the workflow of a cholecystectomy (surgical removal of the gallbladder) is depicted.

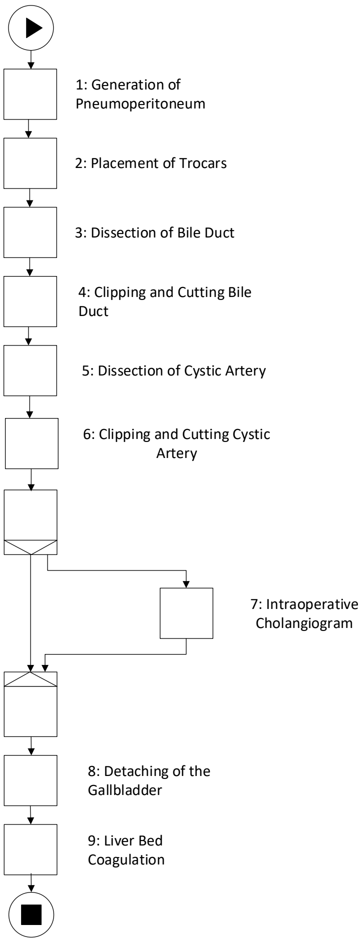
1: Generation of Pneumoperitoneum
2: Placement of Trocars
3: Dissection of Bile Duct
4: Clipping and Cutting Bile Duct
5: Dissection of Cystic Artery
6: Clipping and Cutting Cystic Artery
7: Intraoperative Cholangiogram
8: Detaching of the Gallbladder
9: Liver Bed Coagulation

Figure 4.1: Workflow of a cholecystectomy (surgical removal of the gallbladder) in YAWL. A sequence of tasks is followed by a decision, and two additional tasks after the two possible flows are merged.

The sequence of tasks is as follows: First, carbonic acid gas is injected to inflate the abdomen (stomach). In task two, trocars, sharpened tubes, are used to break through the abdominal wall. Trocars can be used to enable the placement of additional medical instruments during the surgery. In the third task of the workflow, the bile duct is dissected (exposed). The following task comprises the clipping and cutting of the bile duct. In task five the cystic artery is dissected in preparation for the clipping and cutting in task six. After this task, an intraoperative cholangiogram (radiographic imaging of the bile ducts with contrast medium) is optionally performed. Thus, a decision has to be made after task six. When the two possible flows are merged, another sequence of tasks has to be accomplished. It includes the detaching of the gallbladder and the liver bed coagulation.

Each of the tasks is represented by a set of characteristic features. In this work, we utilize surgical instruments to characterize the single tasks. The presence of a certain surgical instrument is given as observation for our phase detection models.

5 Combining YAWL and DBNs

To provide a reliable phase detection, we propose the combination of Yet Another Workflow Language (YAWL) with Dynamic Bayesian Networks (DBNs). YAWL is used to model the relationship between surgical phases, in the sense that possible transitions from one phase to another can be specified. The upper part of Figure 5.1 shows the YAWL model of an exemplary workflow with three subsequent phases a, b and c. The workflow starts with an input condition (black triangle in a circle) followed by the three tasks a, b, c (squares), and ends by a final condition (black square in a circle). A frame comprising the current phase and the subsequent phases is shifted while the workflow progresses. Depending on this frame, a different DBN model is chosen.

In order to do so, the corresponding DBN models have to be linked with the state transitions of YAWL. In Figure 5.2 the internal structure of an atomic YAWL task is outlined. The notation is inspired by Petri Nets [Pet62], and assumes that there are so called transitions which consume marks, so-called tokens. Additionally, there are so called places where these tokens can be stored. Thereby, transitions can be interpreted as internal actions and places can be interpreted as internal states of a task t. The example in Figure 5.2 depicts a transition named *enter* which removes all tokens that enable the task t. In the simple case where task t has no join behavior (e.g. there is only one predecessor task a before task b like in Fig. 5.1), there is only one token (resulting from the predecessor task)

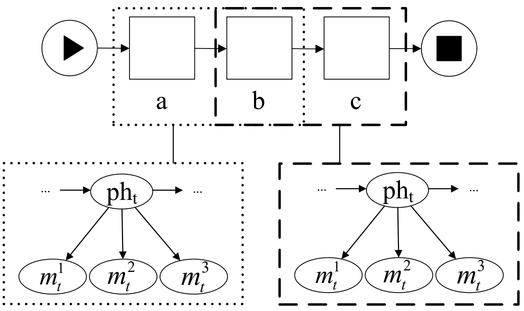

Figure 5.1: A combination of YAWL with DBNs is proposed in this work. YAWL symbols are depicted in the upper part of the figure. The workflow starts by an input condition (black triangle in a circle) followed by three tasks (squares) and ends by a final condition (black square in a circle). In the lower part of the figure, DBNs are shown. Each DBN is associated with a frame comprising the current task as well as possible follow up tasks.

to remove respectively. If there is join behavior and $f_{\text{join}}(t)$ = AND, all input places of the transition $enter_t$ need to hold a token. I.e., task t can only be entered iff all other predecessor tasks are exited. For $f_{\text{join}}(t)$ = XOR and $f_{\text{join}}(t)$ = OR, different rules for joining apply. Here, at least one token has to be present. For more detailed information please refer to [AH03]. Further details on Petri Nets can be found in [Rei13].

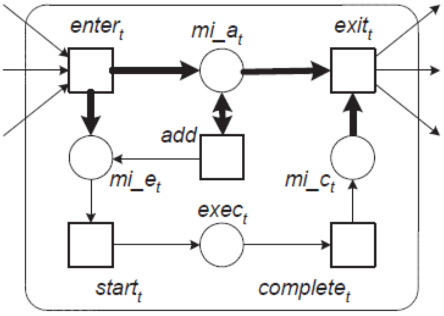

Figure 5.2: The internal states of an atomic YAWL task can be represented by a Petri Net inspired notation. Modified from [AH03].

In Figure 5.2, the places mi_a_t, mi_e_t, mi_c_t, and the transition *add* are necessary for managing multi-instance tasks. For simplification, we consider a task to be a single instance. That means one and the same task cannot be simultaneously executed more than once.

Once, the transition *enter*$_t$ has produced one token, the transition *start* is enabled and can occur. When the superordinate task t starts to be executed, the internal transition *start* consumes the corresponding token. While task t is executed, a token resides in place *exec*. As soon as the superordinate task t is completed, the transition *complete* consumes this token and produces a new token in the adjacent place. The tokens produced by transition *exit* can activate follow up tasks. Different activation mechanisms are possible: 1.) No split behavior, i.e., only one token allows to activate a subsequent task. 2.) In case of an AND-split, for each subsequent task, one token is produced. 3.) In case of XOR- or OR-split at least one token is produced for all of the successors.

In our approach that combines YAWL and DBNs, we chose the appropriate DBN model based on the current phase. The current phase is represented by the task currently executed, i.e., a token is present at the task's internal place *execute*. Each task is linked to a unique DBN model. This model consists of a root node containing the current task, as well as all follow-up tasks as possible outcomes. The follow-up tasks are given by the successors of the task's internal transition *exit*. All other tasks of the network are not considered for classification in this case. As soon as the DBN model predicts that a follow-up phase is present, the current YAWL task is exited and the predicted successor is executed. Once this happens, a new DBN model is activated and used for the following classification. The process can be described by sliding a classification window over the workflow (cf. Figure 5.1).

6 Verification

To verify our approach, we used expert-based modeled DBNs of a cholecystectomy [PFBss]. We generated 100 surgical procedures with simulated feature values, using a median of algorithmic accuracies of 92%. Out of the generated surgeries, 45 surgeries involved an intraoperative cholangiogram. The proposed combination of YAWL and DBNs was utilized to classify the generated observation sequences. Figure 6.1 depicts the resulting confusion matrix. The results show that the model is able to predict the correct task in most cases (overall accuracy of 89%). There is some confusion (which is expressed by the false classified off-diagonal elements, since they are mistakenly confused with other classes), especially in *Clipping and Cutting Cystic Artery* (task 6) and *Intraoperative*

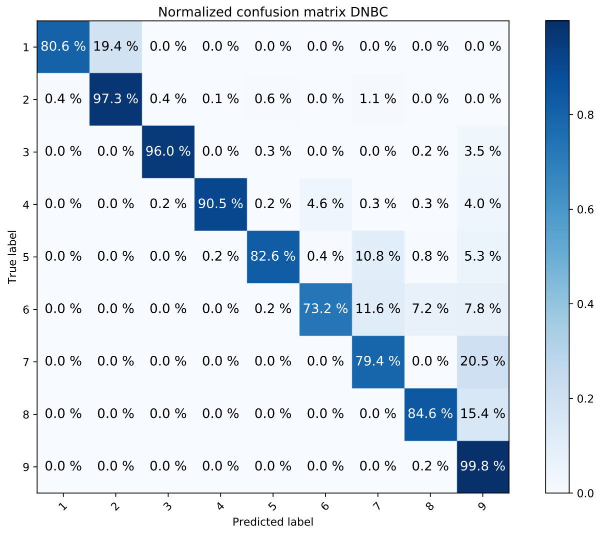

Figure 6.1: Normalized Confusion Matrix. A row represents an instance of an actual surgical step, whereas a column represents an instance of the predicted surgical step. Consequently, the values of the diagonal elements represent the degree of correctly predicted classes.

Cholangiogram (task 7). In the first case, the observations are predominantly confused with the two possible follow-up tasks (*Intraoperative Cholangiogram*, task 7, and *Detaching of the Gallbladder*, task 8). Further confusion is present in *Intraoperative Cholangiogram*, but predominantly with the last phase 9. This is, because the model can hardly compensate wrong predicted transitions, which is a direct consequence of the frame-wise moving classification window. To compensate this aspect, auxiliary backward transitions, e.g. to the previous task, could be added to the model. To sum up we can say that the proposed models work as expected. Nevertheless, there is room of improvement concerning the robustness of the classification which will be addressed in upcoming publications.

7 Conclusion

In this work, we discussed the fundamentals of YAWL (Yet Another Workflow Language) and DBNs (Dynamic Bayesian Networks) in detail. On that basis, we introduced a combination of YAWL and DBNs for a surgical phase detection on the application example of the removal of a gallbladder (cholecystectomy).

We showed that the proposed models are able to predict most of the tasks of the surgery correctly. Nevertheless, there is some room of improvement with respect to the robustness of the classification which will be adressed in future research.

Detecting the progress of an on-going surgery plays an important role for context sensitive assistance: A reliable detection of the current task of a surgical workflow is the basis for a tailored decision support for medical practitioners. Subsequently, an intraoperative support can help to prevent errors, injuries, negligence or malpractices.

Bibliography

[AH03] W Aalst and A Hofstede. Yawl: Yet another workflow language (revised version). Technical report, Technical Report FIT-TR-2003-04, Queensland University of Technology, Brisbane, 2003.

[DK89] Thomas Dean and Keiji Kanazawa. A model for reasoning about persistence and causation. *Computational intelligence*, 5(2):142–150, 1989.

[FL92] E Field and K Lohr. *Guidelines for Clinical Practice: From Development to Use*. National Academies Press, 1992.

[HRAA10] Arthur Hofstede, Nick Russell, Wil Aalst, and Michael Adams. *Modern Business Process Automation: YAWL and its Support Environment*. Springer, Berlin, 2010.

[KF09] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[KWN+15] HG Kenngott, M Wagner, F Nickel, AL Wekerle, A Preukschas, M Apitz, T Schulte, R Rempel, P Mietkowski, F Wagner, et al. Computer-assisted abdominal surgery: new technologies. *Langenbeck's Archives of Surgery*, 400(3):273–281, 2015.

[LJ14] Florent Lalys and Pierre Jannin. Surgical process modelling: a review. *International journal of computer assisted radiology and surgery*, 9(3):495–511, 2014.

[Mur02] Kevin Patrick Murphy. *Dynamic bayesian networks: representation, inference and learning*. PhD thesis, University of California, Berkeley, 2002.

[Pad10] Nicolas Padoy. *Workflow and Activity Modeling for Monitoring Surgical Procedures*. PhD thesis, Université Henri Poincaré-Nancy I, 2010.

[Pet62] Carl Adam Petri. *Kommunikation mit Automaten*. PhD thesis, Universität Bonn, 1962.

[PFBss] Patrick Philipp, Yvonne Fischer, and Jürgen Beyerer. Expert-based probabilistic modeling of workflows in context of surgical interventions. *CogSima 2017, IEEE Conference on Cognitive and Computational Aspects of Situation Management*, 2017, (in Press).

[PFHB16] Patrick Philipp, Yvonne Fischer, Dirk Hempel, and Jürgen Beyerer. Framework for an interactive assistance in diagnostic processes based on probabilistic modeling of clinical practice guidelines. In *Emerging Trends in Applications and Infrastructures for Computational Biology, Bioinformatics, and Systems Biology*, pages 371–390. Elsevier, 2016.

[Phi16] Patrick Philipp. Framework for modeling medical guidelines based on the translation of uml activities into yawl. Technical report, KIT Scientific Publishing, Karlsruhe, 2016.

[Rei13] Wolfgang Reisig. *Understanding Petri Nets*. Springer, Heidelberg, 2013.

[SGM$^+$11] Earl Steinberg, Sheldon Greenfield, Michelle Mancher, et al. *Clinical Practice Guidelines We Can Trust*. National Academies Press, 2011.

[VvdAtH06] HMW Verbeek, Wil MP van der Aalst, and Arthur HM ter Hofstede. Verifying workflows with cancellation regions and or-joins: an approach based on relaxed soundness and invariants. *The Computer Journal*, 50(3):294–314, 2006.