Advance Access publication on 3 May 2021 doi: 10.1093/iwcomp/iwab011

# Using Task Models to Understand the Intersection of Numeracy Skills and Technical Competence With Medical Device Design

Judy Bowen[1,*] and Diana Coben[2]

[1]*The University of Waikato, Hamilton, 3240, New Zealand*
[2]*King's College London, London, WC2R 2LS, UK, and University of East Anglia, Norwich, NR4 7TJ, UK*
[*]*Corresponding author: jbowen@waikato.ac.nz*

**Task models are used in many different ways throughout the design and development of interactive systems. When the interactive systems are safety critical, task models can play an important role in ensuring system behaviours are consistent with user requirements, which may help to prevent errors. While task models can be used to describe a user's goals and the steps required to achieve that goal, to understand where user errors may occur we also need to consider the users' understanding of how to perform a task and how this relates to the system they are using. Our focus is on the use of medical devices such as syringe drivers and infusion pumps for intravenous medication, which forms a major part of hospital inpatient care throughout the world. While we might rely on software engineering and human factors techniques to ensure correctness of such devices, their use by medical personnel in practice includes other factors that are equally important. These include training medical personnel in the use of medical devices. Also numeracy education for medical staff to ensure that they are able to set up and perform the necessary calculations to convert prescribed medication into the appropriate values and measures for their delivery mechanisms. We have developed an approach that aims to bring together concepts of technology design (both functional correctness and usability concerns), numeracy and medication delivery competency. In order to do so we use task models as a common language that enables us to consider these different domains in a single way. We find that the ability to describe the two domains within a single process allows us to compare models of knowledge, tasks and use of devices, which can elicit potential mismatches and problems.**

RESEARCH HIGHLIGHTS

- Task models are presented as a common notation for describing numeracy competence and selected parts of safety critical systems and their use.
- The development of task models of different types is described and related to interactive system design processes.
- A process for combining and comparing the task models to identify mismatches in user knowledge and device use is presented.

*Keywords: task models; technical competence; numeracy; safety-critical systems*

*Handling Editor: Regina Bernhaupt*

*Received 18 September 2019; Revised 1 June 2020; Accepted 10 March 2021*

## 1. INTRODUCTION

Safety-critical interactive medical systems, such as infusion pumps and syringe drivers, are used in a variety of medical settings all over the world. These types of devices are used to deliver medication, fluids, etc. to patients and as such they are relied upon in a number of different medical settings for a diverse range of uses (from pain medication to life-saving drug delivery). Safety-critical interactive systems in general are systems or devices that involve user interaction and that have the potential to be hazardous if they malfunction or if user errors occur. Infusion pumps and syringe drivers have been implicated in numerous adverse events (i.e. injuries 'resulting from a medical intervention'; Kohn *et al.*, 2000), the results of which range from minor patient inconvenience through to serious harm or even death [FDA, 2010]. The US Food and Drug Administration has developed a project aimed at improving this situation (the Generic Infusion Pump project[1] ), which focuses on the design and development of medical technology. This work, and similar research in this domain, is based around either software engineering—seeking to ensure the devices we build behave correctly at all times [Campos & Harrison, 2011] or usability research—seeking to improve the design so that devices are easier to use [Thimbleby, 2015]. However, there are alternative, complementary approaches in other disciplines, which have the same goal of reducing medication delivery errors. These include training medical personnel in the use of medical devices [Vipond, 2016] and also with regard to the numeracy skills required to calculate medication dosages correctly for different delivery mechanisms [Coben & Weeks, 2014].

When we focus on 'user error' as a technological problem we make assumptions. For example, if an error is made leading to the wrong amount (volume) of medication being delivered to a patient, we may assume that the medical professional setting up the infusion (whom we henceforth refer to as the 'user') has started with the correct values for setting up an infusion and made an error in setting up the device or entering the required values. Indeed, number-entry errors are a common occurrence in this domain [Thimbleby, 2015]. However, in practice, it may be impossible to determine at which point the error has actually been made. It may be that rather than a number entry error, it is in fact a calculation error in which the user starts the infusion set-up with wrongly calculated values that they subsequently enter into the device correctly.

Numeracy education for medical staff focuses on ensuring that they are able to set up and perform the necessary calculations to convert prescribed medication into the appropriate values and measures for their delivery mechanisms, which may be tablets of a given strength (e.g. 5 mg) or liquid solutions containing given amounts of medication (e.g. 20 mg per 2 ml). It is essential that this precursor to actual medication delivery is correct and that the medical personnel can then safely and correctly deliver the correct medication doses using the technology provided.

In a previous work [Coben & Bowen, 2019], we considered how we might bring together concepts of technology design (both functional correctness and usability concerns), numeracy and medication delivery competency and outlined the groundwork needed to achieve this. We considered the use of task models in the domains of medical device design and how we might extend this to the context of the development of competency for nurses. Task models seemed an appropriate choice due to their prevalence in interactive system engineering and their roots in human cognitive processes. We build on this here by describing the process for creating such task models and combining them with models of user tasks with devices.

Task models emerged from the discipline of cognitive task analysis, which has its roots in the field of human factors and ergonomics. Their use in human computer interaction (HCI) processes that focus on the design and analysis of interactive systems has become commonplace, and the concept of task is central to this. People use interactive applications to perform tasks; therefore, the use of task models supports a user-centred development process. A variety of task analysis and modelling methods have been developed, and while there are differences in their intended use, notation and expressiveness, most are influenced by ideas of the hierarchical task analysis [Annett & Duncan, 1967], which enables decomposition of higher level tasks into smaller steps until we can define the task as a hierarchy of actions.

Task models in interaction design can be used to describe user goals and requirements and are often seen as complementary to dialog models, which express the available commands and behaviours of systems. We further exploit this by considering task models of systems (by which we mean the way in which a user performs tasks using a given system), which act as a complement to user task models. The task models of a system describe *how* a task may be achieved using the system, with the hierarchical steps describing groups of allowable actions that can be performed when interacting with the system. We describe this in more detail in section 2.2.

In this paper we demonstrate how we can identify common properties (tasks and knowledge required to complete those tasks) across a variety of different task models. While the 'human' aspects of our work relate to all medical personnel, in this paper we focus on nurses because they are most likely to deliver medication directly to the patient: as such, they are 'the last line of defence' in medicines management in the healthcare context [Leufer & Holdforth-Cleary, 2011]. Similarly, there are many medical devices in use for all sorts of purposes within healthcare settings, but here we focus on infusion pumps and syringe drivers specifically.

We focus on the domain of use of medical devices, such as infusion pumps and syringe drivers, and consider including

[1] https://rtg.cis.upenn.edu/gip/

task models that describe the goals of the users of medical devices, task models derived from the devices themselves (as above) and task models describing user competence requirements. This third category of task model describes the required knowledge and technical competence that is required to set up the necessary calculations required as a pre-cursor to the use of the medical device.

Our work suggest a new way of facilitating knowledge transfer between numeracy education and medical device design and usage, using task models. We aim to support medical professionals' and students' numeracy education as well as to inform the design of medical devices based on a better understanding of the use and potential errors of medication delivery by trained professionals.

The contributions of this paper are the description of the different types of task model we use to provide a 'common language' across different domains' and a demonstration of the applicability and usefulness of generating and comparing these different types of task models. We demonstrate a new way of considering related information from different domains with the goal of providing benefits to both.

In the next section we describe the background to our work and discuss related work in the domains of task modelling and interactive system design. We also introduce the concept of competency in medication dosage calculations, which is used as the basis for numeracy education for nurses. In section 3 we introduce the task models we will use for device behaviour. We show how they can be derived from models of a system and how they relate to standard task models that describe the users' needs. We extend this in section 4 by creating task models derived from a medical numeracy education tool, safeMedicate®. In section 5 we demonstrate ways of comparing and combining the different variants of task models proposed. This is followed by a discussion of what has been achieved and how this might be used in different ways in future research. We finish with conclusions and outline the next steps we are planning for this work.

## 2. BACKGROUND AND RELATED WORK

### 2.1. Model-based design for interactive systems

Model-based design and engineering for safety-critical software and hardware (such as the medical devices we discuss here) take many forms and can be considered across two different dimensions. The first dimension is the type of model and where it falls on the scale of formality—from fully formal approaches, such as the use of formal specifications, verification, refinement, etc. [Blandford *et al.*, 2011, Bowen & Reeves, 2007, Harrison *et al.*, 2017], to the informal methods of user-centred design and more 'agile' design approaches [Blandford *et al.*, 2010] as well as everything in between [Calvary *et al.*, 2001, Seffah *et al.*, 2009]. The second dimension is how, and where, the models are used: at the design stage before any implementation occurs; as part of the final testing and sign-off of the implemented solution; or throughout the design process.

Other research approaches seek to consider the ranges of both dimensions by focusing on the integration of formal specification techniques (ideally suited in safety-critical domains) with user-centred design approaches (ideally suited for interactive systems) [Bowen & Reeves, 2006]. This allows consideration of both design and functionality using formal and informal methods at the requirements stage and throughout development. It also lends itself to integrated testing approaches as well as post-implementation and reverse-engineering analysis methods [Bowen, 2015, Bowen & Reeves, 2008]. While much of this work does not explicitly incorporate task models, they are usually considered as implicit inputs into the user-centred design approach that forms the basis for UI and interaction models.

Here we assume that for the medical devices we are interested in a model-based approach is followed and that both informal and formal design artefacts are made use of at all stages from initial requirements elicitation through to implementation and testing. Later we will discuss how such models can be developed from, or used to develop, task models for specific systems.

### 2.2. Task analysis and task models in interactive system design

Task analysis is aimed at understanding how a user completes a defined task. It allows us to analyse what a person is required to do to achieve a certain goal (the task) as well as analyse the effort (both cognitive and physical) required to do this. There are a large number of methods, notations and tools used for task analysis within both computer science and psychology (where the origins of task analysis can be found in applied behaviour analysis). The choice of which to use typically depends on the formality of the design process and the use of the task model within that process.

Task models are used in interactive system design in a variety of ways: to help elicit user requirements; to provide a way to model user goals; to analyse cognitive and action loads of achieving goals, etc. Used in this way, task models have evolved from their origins in psychology, where they were used initially to decompose tasks into hierarchically structured sub-tasks of observable behaviours. Task models can also, therefore, be seen as a fundamental way of linking interactive system design with user behaviours and activities. Accordingly, not only are they ideal for modelling and understanding user tasks independently from computational systems but also they can be linked to system design models via a common semantics and hierarchical structure.

While basic hierarchical decomposition may be sufficient in early stages of development to support understanding of user requirements, and proved popular when task analysis began to be incorporated into the domain of HCI (see Shepherd,

1989, for example), the use of task models has evolved within interactive system design and development to include goal-based analysis and conceptual models that are used at various stages throughout the design life cycle. The work by Paternò et al. has extended this further to a comprehensive notation for both task and dialog modelling based on concur-task trees (CTT) [Paternò *et al.*, 1997], which includes a variety of logical and temporal operators for ordering and iteration, as well as tools to support creation of, and reasoning about, such models [Mori *et al.*, 2002].

Palanque et al. have incorporated these CTT into a petri-net based modelling and development environment for use within safety-critical interactive system design [Barboni *et al.*, 2010, Martinie *et al.*, 2013]. These types of extensions, and others (e.g. [Dittmar & Forbrig, 2003]), allow the use of straightforward hierarchical models for complex reasoning about safety, user collaborations, reconfigurable human–machine interfaces, etc. As such, their work demonstrates the flexibility of task modelling and the ability to incorporate it into formal modelling for a wider range of uses. This supports the goals of our work and motivates the use of task models in the manner we propose.

Of particular relevance here is the use of Palanque et al.'s petri-net environment to support training of operators of safety-critical systems. This takes a similar approach to our own, by using task models to link specific device models to another domain (in their case, training procedures and programs) [Martinie *et al.*, 2011]. While their aims are different in that they seek to develop appropriate training programs by integrating task models within the simulation environment PetShop [Palanque *et al.*, 2009], the use of taskmodels to bridge cross-domain knowledge demonstrates their applicability in such approaches. In [Martinie *et al.*, 2013], Martinie et al. extended this work to include knowledge objects and information into the models to add descriptions of what information users may require and different representations of knowledge. While we do not go into such detail here this work is clearly relevant, and we discuss this again later in section 6.

Integrated tools and simulation environments such as PetShop provide the ability to incorporate different concepts and approaches (in this case, task models, interactive system models and training procedures) into a single environment. However, the downside of this is that in order to take advantage of such a tool, everything has to be modelled and developed within this one tool. Frequently, the diversity of artefacts within interactive system development and the heterogeneity of different groups within the design team means this is not always the most suitable choice. This is addressed in the work by Forbrig *et al.* [2014], which seeks to create a more lightweight approach through the use of sub-models, sub-routines and generic components. This use of composition is similar to our proposed connection of corresponding models, which we discuss in section 3. For our work here we take the simplest approach to describing task models hierarchically and do not assume the use of any particular existing notation or tool. Our intention is to use task models in a 'lightweight' manner as one part of the development process, that is we consider then a tool that we may deploy for particular aspects of understanding rather than for the entire development process.

Unlike the approaches seen in 'Instructional Task Analysis', which use task decomposition as a means to decide what skills and knowledge is required by users of a particular system, here we look at the skills and knowledge of clinicians required to administer medication and compare them with the tasks of using technological systems to deliver such medication.

### 2.3. Modelling users

Although task models describe user behaviours, these are at a level of the actions required to achieve a goal. That is, they typically assume correct or optimal behaviours. Comparisons between such models and actual user behaviours in practice can be used to identify, and even model, errors based on missteps or slips (see Johnson, 2011, for example), but task models alone do not identify such errors. More frequently they might be used as part of approaches such as 'key-stroke level models' [Card *et al.*, 1980] and 'goals, operators, methods and selection' [Card *et al.*, 1983], which seek to identify cognitive load and effort, which in turn may suggest potential for error.

In contrast, work that does seek to model user cognition and identify potential errors based on this makes different types of assumptions. Blandford et al. have focused on the effect cognition has on user behaviour [Blandford *et al.*, 1997] and extended this to consider distributed cognition for use with multi-user systems [Blandford & Furniss, 2005]. More recently, they have looked at the effect distributed cognition has on medical practices [Berndt *et al.*, 2015, Rajkomar & Blandford, 2012] both in clinical settings and in the home. We also consider the work by Curzon et al. who use salience as an important property in understanding cognition and the effect this has on interactive system design [Ruksenas *et al.*, 2008]. These examples categorize different types of causal effects (distributed behaviours, salience) and their potential to lead to error and then use these to either improve processes or improve the design of safety-critical systems. Rather than consider user behaviours based on such concepts, we instead focus on the driving factor, the key knowledge that users have (or should have) prior to undertaking particular tasks and the competence they demonstrate in performing these tasks. So, at this stage, we do not incorporate error or deviations from the prescribed knowledge in our models.

We base our models of user actions on expected knowledge provided by the numeracy training. This professional knowledge base has been synthesised in the European Qualifications Framework for Lifelong Learning as 'The proven ability to use knowledge, skills and personal, social and/or methodological abilities, in work or study situations and in professional and personal development' [European Education and Culture

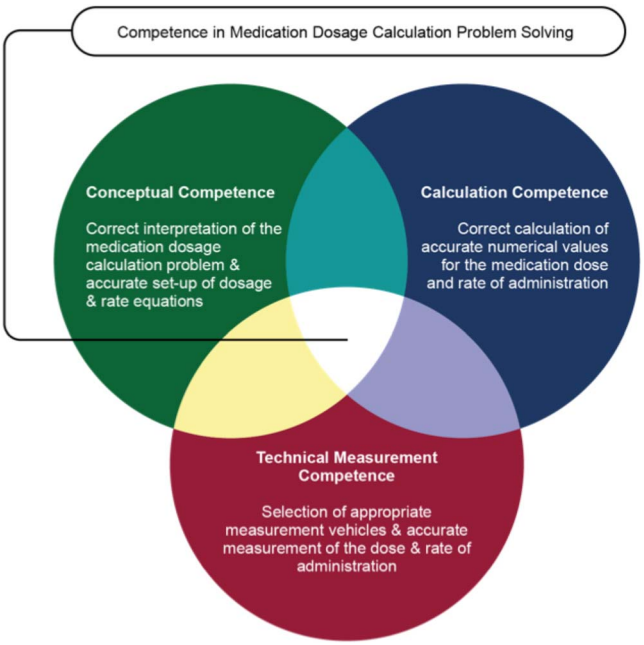


**FIGURE 1.** MDC-PS competence model [Weeks *et al.*, 2013a].

Commission, 2008]. This notion of professional clinical competence underpins our research. We discuss this next in the context of medication dosage calculation problem solving (MDC-PS) in nursing.

### 2.4. Competence in medication dosage calculation problem solving in nursing

Weeks et al. [Weeks *et al.*, 2013a] have proposed a competence model for MDC-PS, which represents the intersection between the ability to interpret the dosage calculation problem and accurately set up rate equations (conceptual competence), the correct calculation of accurate numerical values for the dose and rate of administration (calculation competence) and the selection of appropriate measurement vehicles and accurate measurement of the dose and rate of administration (technical measurement competence), see Fig. 1 [Weeks *et al.*, 2013a].

Specifically, the MDC-PS competence domains are characterized as follows:

(i) Conceptual competence refers to the need to:

  (A) Understand the elements of prescription charts, dispensed medication labels and medication data sheets and monographs, and to subsequently extract the numerical information necessary to set up the dosage problem correctly.
  (B) Position the numerical information appropriately and correctly in an equation format for calculation.

(ii) Calculation competence refers to the need to correctly apply arithmetical operations and compute an accurate numerical value within a safe and acceptable tolerance range for the prescribed medication dose and/or rate of administration.

(iii) Technical measurement competence refers to the need to:

  (A) Select an appropriate medication administration measurement vehicle (tablet or capsule, oral liquid medicine measurement cup, syringe, infusion pump, etc.)
  (B) Accurately transform the calculated numerical value to the context of the measurement device/formulation and measure the correct dose of prescribed medication; and/or administer the correct rate of prescribed medication/IV infusion fluid.

An uncorrected error in any one or more of these will result in a medication dosage error in the practice setting [Weeks *et al.*, 2013a]. Technical measurement competence is particularly relevant to our focus in this paper since it involves the use of medication delivery devices.

## 3. TASK MODELS OF USERS AND MEDICAL DEVICES

### 3.1. Task modelling approach

In this work we describe task models from different perspectives. Firstly we use the standard approach of describing user goals in a hierarchical manner to create a typical user task model. Secondly, we create device-specific task models that describe how a task can be achieved using a specific medical device. Thirdly, we create numeracy calculation task models that describe how medication dosage calculations should be performed in order to correctly set the dosage for a medical device. These are user task models, but with the addition of knowledge requirements. In this section we describe how we develop and use the first two of these task models, and then in section 4 describe the third type and show how we relate all three together.

In addition to the different types of task model described above, we also consider each of these under one of three categories. The first is the *generic* model. This is the description of a task in its most general case with no specific details beyond the higher-level goal. For example a generic task model may describe a task such as ‘Set up and start infusion’. There is no more detail provided, it is the most general case of the task. The second category is the *specialized* model. A specialized task model for a user may be ‘Set up a medication delivery based on the prescription of patient X using a syringe driver’. A specialized task model for a device may be ‘Set up a T34 syringe driver to deliver medication’.

The final category is a parameterized task model. Parameters in these task models are shown as [$P_n$] in the model and

represent actual values (and units where appropriate) for variables, for example 'Set volume of medication to be infused to 200 ml'. This would be indicated in the model as [$P_1$] where $P_1$ = 200 ml. On smaller models the actual values may be shown rather than the [P] indicator, which is used to constrain the visual size of the model. The parameters provide the detail for fulfilling the tasks/steps and as such we might consider them as guards. Parameters may be values, values with units, calculations, conditions or any other variables that apply to the steps within the model. Rather than a description of a predicate (which is more common when we think of guards on fulfillment of steps) they are intended to represent some parameter which is meaningful to the task. For example an action 'set up equation' with a parameter [ $P$ = 25.3 mcg x 10 kg x 30 mins] describes that the fulfillment of setting up the equation would be when it matches the parameter.

Our focus is not on describing a new task modelling language or notation, rather finding the simplest way to describe the types of tasks we have described above. In section 2.2 we discussed several existing notations and supporting tools used in interactive system engineering. These are designed to support a full development process, and as such may be used/adapted for our work in the future, but here to retain the simplicity for describing our process we will simply use a hierarchical structure to show the decomposition of tasks with the minimal set of operators required for description of our examples. These operators are as follows:

`A - > - B`

Ordered sequence, A must be completed before B starts

`A - || - B`

Unordered sequence, A and B in any order

`A [p]- > - B`

Ordered sequence with parameter, A must be completed and *p* satisfied before B starts

`A [p] - || - B`

Unordered sequence, A and B in any order until *p* is satisfied. Iteration is assumed as necessary and we do not consider temporal properties.

These are defined for descriptive purposes of our examples and it is not our intention to give semantics for these, they could be easily replaced with any suitable existing notation with the same operands. The final convention we adopt is the use of a dotted rectangle surrounding a sub-task(s) in a user model that has a corresponding device model, and vice versa, these may also be annotated with the name(s) of corresponding model(s). We now give concrete examples of each of these.

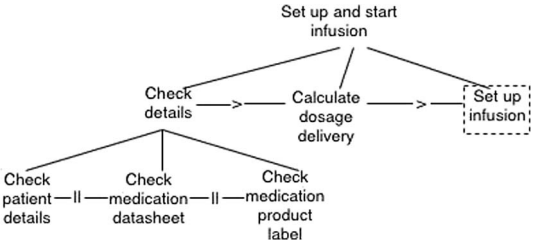


**FIGURE 2.** Generic user task model for setting up an infusion.

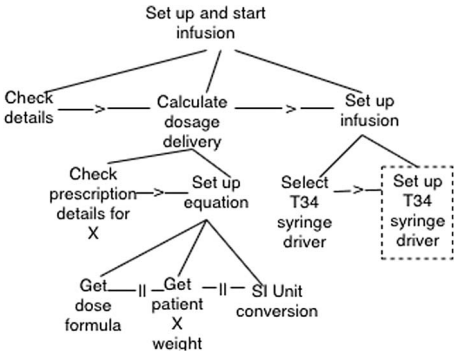


**FIGURE 3.** Specialized user task model for setting up an infusion.

### 3.2. Examples of user and device task models

We start with a generic example of a user task 'Set up and start infusion', which is shown in Fig. 2. The top level goal becomes the root node of the model, which is then hierarchically decomposed into atomic user actions. The top level goal is broken down into three sub-tasks, which are performed in order one at a time, indicated by the > operator. We have further decomposed the 'Check details' sub-task into three smaller sub-tasks, which can be performed in any order, as indicated by the '||' operator. The 'Set up infusion' sub-task is surrounded by a dotted box to indicate that it connects to a corresponding device model. For larger sets of models we might label the dotted box with the name of the corresponding model(s). We could continue to decompose the steps until we reach atomic actions at which point we would have a complete, generic model.

We might then specialise this model, for example by considering a particular type of medication and/or delivery mechanism. From this we might produce the specialised model shown in Fig. 3 (note: we only show some parts of the suggested specialization, which is in keeping with our approach of using partial and smaller models as required).

As a final step we might parameterize the model with values and/or conditions relating to units, for example we could specialize the 'Set up equation' sub-task with a dose formula of 20 ml per kg per minute using a solution of 50 ml/mg. This is shown in Fig. 4 (note: p1 is the combination of all of the other

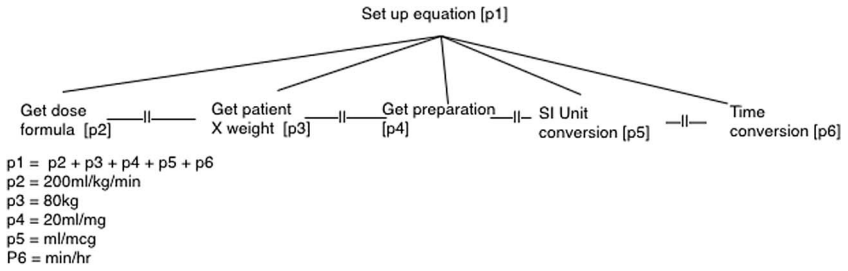


**FIGURE 4.** Parameterized user task model for setting up equation to calculate dosage.

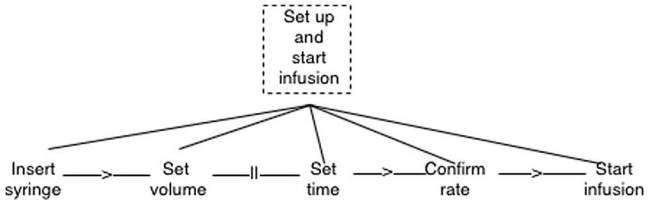


**FIGURE 5.** Generic model for setting up a device for infusion.

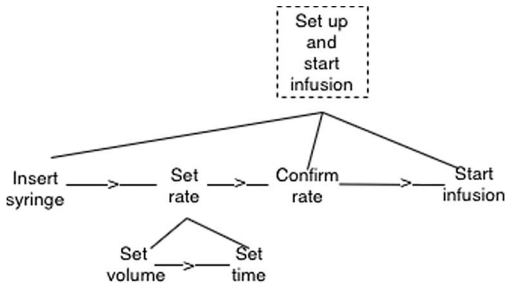


**FIGURE 6.** Specialized model for setting up a T34 syringe driver for infusion.

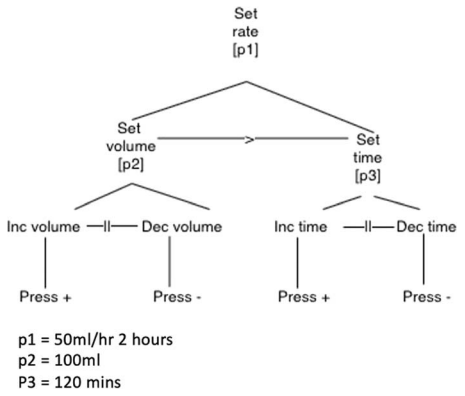


**FIGURE 7.** Parameterized partial device model for setting up a T34 syringe driver with given dosage.

parameters, which is described as p1 = p2 + p3 + p4, etc. This is intended to mean *and* rather than addition).

Similarly we can develop device models to show how the same tasks would be performed using a medical device such as a syringe driver. Figures 5, 6 and 7 show selected parts of a generic device model, a specialized T34 syringe driver model and a parameterized T34 syringe driver model for a task of dispensing 50 ml per hour for 2 hours. In the generic model we do not assume a required order for entering volume to be infused and time (hence Set volume - || -), but in Fig. 7 we must explicitly represent the order mandated by the T34 syringe driver. Hence, we create a new decomposed step *Set rate* and the operator changes to ->-.

### 3.3. Deriving the device task models

One of the advantages of relating the user task model to the device task model is to ensure that a proposed device can support all of the required tasks. In addition, by considering the structures of the task models and the way the hierarchy is defined, we might identify where potential user error or mode confusion may occur. As we have stated, we assume that a model-based development process is being followed, which enables us to generate the device task models from the design models, We describe this next.

As discussed in 2.1, a variety of different models and modelling techniques exist for interactive systems. Here we focus on the presentation model approach described in [Bowen & Reeves, 2017], which uses several different models (both formal and informal) at varying levels of abstraction throughout the design and development process. This enables both formal verification of properties such as safety [Bowen & Reeves, 2013], as well as supporting prototyping and lightweight UI design. As such it provides another example of using models and partial models at different levels of abstraction in a modular way as required, rather than fully describing all parts of a system using a single model (see Bowen & Reeves, 2017, for more details on this). Interface designs can be described by the interactive elements (widgets) of the design and their intended behaviours. As such, a formal structure can be given to the narrative behind prototypes, personas, storyboards, etc., which (among other things) removes ambiguity. Although a


**FIGURE 8.** T34 syringe driver.

full explanation of these is beyond the scope of this paper we introduce the basic elements by way of an example to demonstrate the derivation of the device task models.

Consider the T34 syringe driver shown in Fig. 8. This is a modal device, that is, the behaviour of each of its buttons is dependent on the current mode of the system. A simple presentation model can be constructed, which describes each mode as a collection of the available widgets. These are described in a tuple giving a name to the widgets, their type (whether they generate events, e.g. ActionControls, or respond to events, e.g. Responders) and their intended behaviour. For example, the device has a mode that enables a user to enter the volume of medication to be infused, which we call 'SetVolume'. The presentation model of the 'SetVolume' mode might include:

```
SetVolume is
  OnButton, ActionControl, (I_Init),
  UpButton, ActionControl, (S_IncVolume),
  DownButton, ActionControl, (S_DecVolume),
  Display, Responder, (S_IncVolume,
  S_DecVolume),
  ..
  ..
```

as well as the rest of the widgets, which we omit here for brevity. Each mode of the device is described in a similar way. The button behaviour names are prefixed with either an 'I_', which indicates it is a behaviour relating to interface navigation (mode change) or an 'S_' if it relates to system functionality.

The lightweight presentation model is informal but can be linked to other models, which then give formal meaning and semantics to these simple tuples. For example, the presentation interaction model (PIM) is a state transition diagram with each state representing the presentation model of a mode and transitions showing the behaviours that enable a user to switch between these modes. The PIM gives a formal meaning to the I-Behaviours of the presentation model. Similarly a relation between S-behaviours and a formal specification (typically given using the Z specification language in this approach) provides the formal semantics for those S-behaviours.

From these models of the device we can derive interaction sequences. An interaction sequence is the set of actions and interactions that a user performs with a given system to complete a task [Turner *et al.*, 2017]. As such, they describe actions a user undertakes for a given task from a given device state. For example, if the device is in the 'SetVolume' mode with a current volume value of '0', then setting the volume to be infused to 10 ml requires the user to press the 'UpButton' 10 times, which is represented in the interaction sequence as *ActionWidget[num times]*, e.g.'PressUpButton[10]'. Interaction sequences can be generated automatically from the device models and can be used to represent optimal paths of actions (as in the set volume example here) or may include additional, arbitrary or erroneous actions. Such interaction sequences are, therefore, a type of task model (describing the steps to complete a task), but they are sequential rather than hierarchical and at the lowest level of abstraction.

These types of descriptions are model-specific in that they are explicitly tied to the detail of the formal model, and they pertain solely to the goals of the user in terms of *what* they want to achieve without the low-level details of *how*.

Task models can be used to inform device design because their inclusion in a typical user-centred design (UCD) approach means that the formal models derived from UCD artefacts have this information embodied within them. That is, a prototype or storyboard created to examine initial design ideas is partly based upon the user tasks (as well as requirements, guidelines, safety regulations, etc.).

Model-specific task models, such as interaction sequences, can be used iteratively to help refine design artefacts. For example, our interaction sequence step 'PressUpButton[10]' described above may lead to a design evolution where a long press on the 'Up' button increases a value in increments of 10. This leads to shorter interaction sequences for some tasks, which might be seen as better for user experience and usability. There is, therefore, already a relationship between task models and device models beyond their use as an initial design artefact and we seek to build on that here.

### 3.4. Task models of medical devices

The syringe driver in Fig. 8 is a CME Niki T34. It is used to deliver a pre-determined amount of medication from a syringe to a patient over a pre-defined period of time. The device has eight buttons that the user can interact with, as well as a small screen that provides information and feedback. In order to set up medication delivery the user needs to undertake several different tasks, some using the device (inserting the syringe, setting up the dosage rate, etc.) and some independently from the device (calculating correct volume and time according to the prescribed dose, getting the medication, etc.).

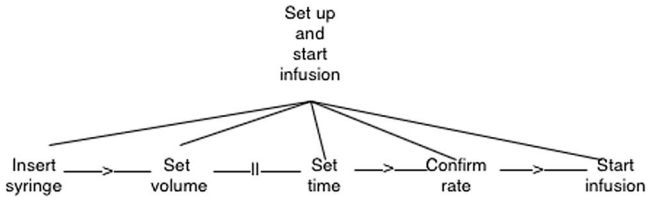


**FIGURE 9.** Task model for setting up and starting infusion.

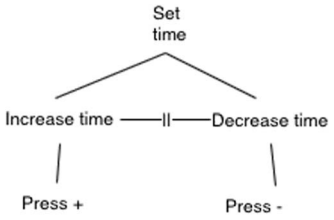


**FIGURE 10.** Task model for setting infusion time.

Each of the tasks can be combined into a single action, 'Set up infusion', which can then be broken down into hierarchical steps in a typical task analysis fashion. So, we can create a model for the generic task that a user would perform with this type of device. If we again consider the task of setting up an infusion, this can be decomposed into the following five sub-tasks:

(i) turn on syringe driver
(ii) insert syringe into driver
(iii) set volume to be infused
(iv) set time for infusion
(v) start infusion

There are a number of assumptions made before this task can be carried out, which are reliant on previous tasks (such as selecting the appropriate syringe type and size and drawing up the medication) having been successfully completed. A full task model would include all of these, but for now we focus on just the task actions relating to the medical device, which is typical when using such models as part of interactive system development and reasoning.

Figure 9 shows the top level view of these tasks and is the same as the generic device model of Fig. 5. If we specialize the model, some tasks can be further decomposed, for example, in Fig. 10 we see the detail of the sub-task required to set the time of the infusion for the T34.

We can continue to specialize this model for the T34 by decomposing sub-tasks down to atomic steps, which require actual button presses using the corresponding widgets of the particular device.

The informal and formal models (as described above) describe an intended device design based on user tasks and subsequent design decisions. We can use the task models generated from these to identify mismatches between user requirements and implemented devices by comparing the task models. For example, if the task model of the implemented device differs in steps/structure from the user task model, we need to identify what impact this may have (does it lead to users' mode confusion that would increase likelihood of error for example). Differences may also suggest a mismatch between user expectations (as embodied in the UI design models) and the actual implementation. Once identified we might use it to inform user training or to warn of potential user error.

Such comparisons can also be useful when considering multiple devices with similar functionality (or the same devices with different firmware). If defined orderings of actions differ between different instances of devices, this is an area that may also lead to confusion or user error and so enables us to flag a potential problem. A specialized device task model can also be decomposed down to the level of the interaction sequence, so the lowest nodes on the tree represent interactions with actual widgets, such as 'press upButton'. This allows us to start combining the interaction sequences of device models with model-specific task models to ensure that they are consistent.

In this work we also wish to consider a comparison of task models for devices with models of user intentions based on numeracy skills and technical competence. We discuss this next.

## 4. TASK MODELS AND NUMERACY/MATHEMATICS EDUCATION

There is an extensive research literature on mathematical task design [Watson & Ohtani, 2015], likewise on mathematical modelling (e.g. Galbraith *et al.*, 2007), but the literature on 'task models' *per se* relates to human–computer interaction in engineering and computer science (e.g. Paterno, 2001) rather than mathematics or numeracy education. We suggest that bringing insights from these fields together could benefit both. In particular, we suggest that such an integrated approach could improve our understanding of the numeracy demands of various user interface scenarios and the best way of ensuring that users can meet these demands, including through both education and training of users and improved interface design of devices.

Our focus is on user interfaces—and users interfacing—with safety-critical medical equipment involving digital inputs and/or reading, recording and interpreting of outputs by users, on the education and training required for this to be done competently, i.e. efficiently, effectively and, above all, safely, and on the implications of this for the design of device interfaces.

In so doing we are bringing together research and ways of thinking developed in different academic disciplines and professional domains, with very different relationships to—and conceptions of—the notion of task modelling. For example, as

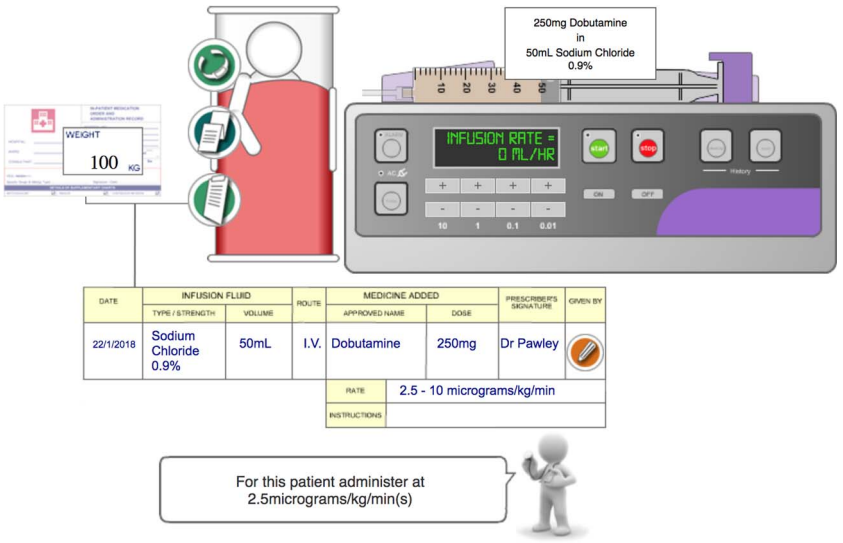


**FIGURE 11.** Screenshot of example problem from safeMedicate® ©Authentic World Ltd.

the editors of a recent book detailing an international study on task design in mathematics education state, 'Task design is at the heart of effective mathematics teaching and learning' [Watson & Ohtani, 2015]. However, 'Despite the recent growth spurt of design studies within mathematics education, the specificity of the principles that inform task design in a precise way remains both underdeveloped and, even when somewhat developed, under-reported' [Kieran *et al.*, 2015].

We suggest that meaningful tasks model activity in the real world in an authentic way [Palm, 2009, Weeks *et al.*, 2013b]. As well as being authentic, we also suggest that tasks should be mathematicallyrich and pitched at an appropriate level of challenge for the learner. The principles of MDC-PS described in section 2.4 have been embodied in an e-learning environment called 'safeMedicate ® ', which has been developed by Authentic World Ltd.[2] This contains authentic clinical dosage calculation problems and supports the development and assessment of competence in dosage calculation problem solving within five skill-based modules. We present examples from these problems to demonstrate the use of task models as a mechanism for structuring the steps and activities users are required to undertake to complete the problems, which therefore represents the steps required to perform the task in a real clinical setting.

Figure 11 shows a screenshot of one of the example problems from the 'Advanced Injectable Medicines Therapy, Continuous Infusion' module.

The problem describes the task of setting up a syringe driver to deliver the described medication. However, before the user can actually perform the set up of the syringe driver, there are

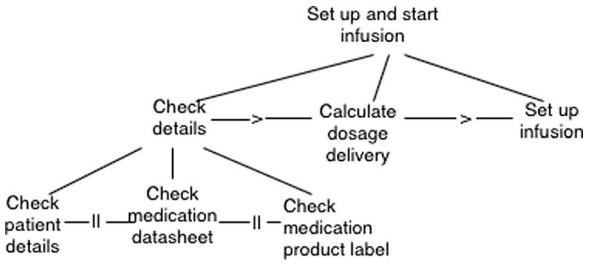


**FIGURE 12.** Task model for dosage calculation and setting up of infusion.

a number of steps they must complete first, such as checking all of the details to ensure that the patient, prescription, medication, etc. are correct and match with each other. Figure 12 shows a partial task model for these initial steps, which is the same as the generic model of Fig. 2.

These could be specialized and decomposed further, as we saw with Figs 3 and 4, using the model answer for the safeMedicate® problem (shown in Fig. 13) to provide the details and parameters. For example we can create the task model for calculating the dosage delivery, as shown in Fig. 14.

The activities described in both the task models based on the numeracy example and those from the device models in the previous section can be linked directly to the numbered steps within the competence model (section 2.4) as follows:

- (i) A is required to check the information and links to the steps under the 'Check Details' sub-task
- (i) B is required to set up the equation with the correct values and calculation steps
- (ii) is required to evaluate the equation
- (iii) A and B are required to set up and start infusion

2 https://www.safemedicate.com/

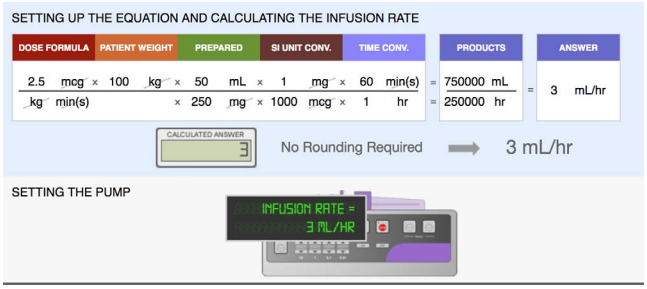


**FIGURE 13.** Model answer screenshot from safeMedicate® ©Authentic World Ltd.

We can, therefore, identify a relationship between the two domains, where fulfilling tasks described in device models depends on competencies outlined in the MDC-PS. The actual relationship depends on the level of analysis (how far we decompose the sub-tasks into smaller steps), but we can see how it starts to become possible to identify errors or mismatches in beliefs, behaviours and requirements as we start to bring the two domains together via the models. For example, if the required technical competence (iii) B (Accurately transform the calculated numerical value to the context of the measurement device/formulation....) is not met, then the user will not be able to accurately set up the medical device for the required dosage.

Now that we have task models that describe tasks for the numeracy competence as well as the technical competence we can begin to combine and compare these models based on their overlapping components (shown in the models of Section 3 with the dotted rectangles), which allows us to consider the intersection shown in the competence model (the central white part in Fig. 2.4), which represents overall competence in medication dosage calculation problem solving. We discuss this further next.

## 5. COMBINING AND COMPARING TASK MODELS

We have described how to generate task models from both interactive system models and numeracy education tasks, and we can now consider how these might be used together alongside traditional task models. We consider how mappings between models, as well as comparisons, can be informative by demonstrating how we might compare and combine the models in a useful manner, i.e. one that is productive of safe, efficient and effective clinical practice. We show three different ways in which this can be used: firstly to identify potential mismatches between user knowledge (from training tasks) and device set up; secondly to inform device design; thirdly to improve training materials and simulations to represent real-world technical artifacts.

By considering the models of user numeracy tasks and device models, we can identify specific mismatches in expected user behaviours and device usage. If we consider a parameterised user task model, such as the example given in Fig. 4 and a corresponding device model, we may find inconsistencies in the parameters (either in missing parameters or conflicting values), which would indicate a potential problem when the device is used. We can either combine models to create a single large model or examine the two individual corresponding models. Figure 15 is a combined model that joins a description of calculating the dosage for a medication delivery with the steps required to set up a specific device to deliver the medication.

We can see that in Fig. 15 there is a parameter mismatch between the units of P2 (unit = hours) and P6 (unit = minutes). While this may seem trivial, it is evident from the recent Graseby incidents (where devices that delivered medication over 1 hour were confused with devices that delivered medication over 24 hours) [Bodkin, 2021] that such errors are both common, and potentially fatal. By identifying such a mismatch we can draw attention to a potential knowledge gap that ensues from the training task when it is applied in the real world to a device that is different from that used in the safeMedicate® tool.

We could also use this information to inform device design. By considering the combined task models during the development process of a syringe driver we could identify the mismatch that occurs and suggest alternative designs that would prevent this. For example we might prototype a device that enables the user to enter the values as calculated but also define the units (ml/hr) of their values. The device might then perform the conversion to whichever values it requires for its own delivery calculation. Even better, if we are using the models in initial design and prototyping processes, we might suggest a version of the device that could be used to evaluate the equation itself. The user would enter all of the values and units from the equation directly into the device, which would then calculate delivery rate and time. This has the added advantage of removing any additional calculation device (such as a pocket calculator or phone calculator), which might be used to evaluate the equation and which has the potential to introduce a whole new set of errors (see Thimbleby, 2000, 2015, for a full discussion of this problem). Because we can tightly couple the design prototypes (via their presentation models) to task models for the designed device, we are able to iterate through this process and move between different levels of abstraction and models as we make changes indicated by any further mismatches we find. This adds another layer of information to this process, as rather than just considering the user task model as a series of required steps to achieve their goal, we can add in the task model of the knowledge-based approach the user will take based on the learning tasks from the numeracy training software.

As a third step we might then also feed this information back to the training providers to inform development of the exercises and simulations to more closely match the real-world scenario. At the moment, safeMedicate® uses an interface for medication delivery devices, which does not represent all types of input

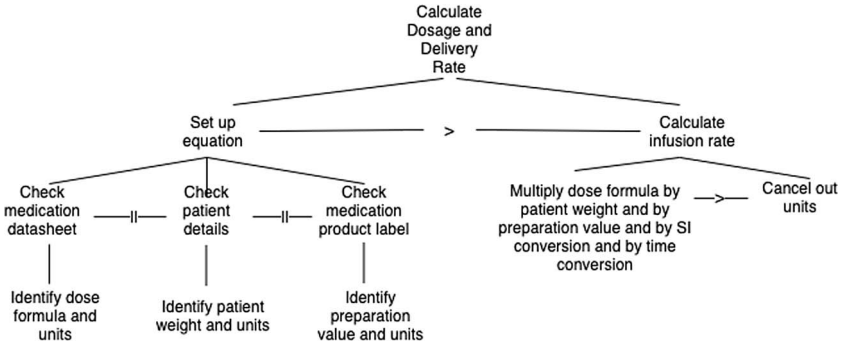


**FIGURE 14.** Task model for calculating dosage delivery.

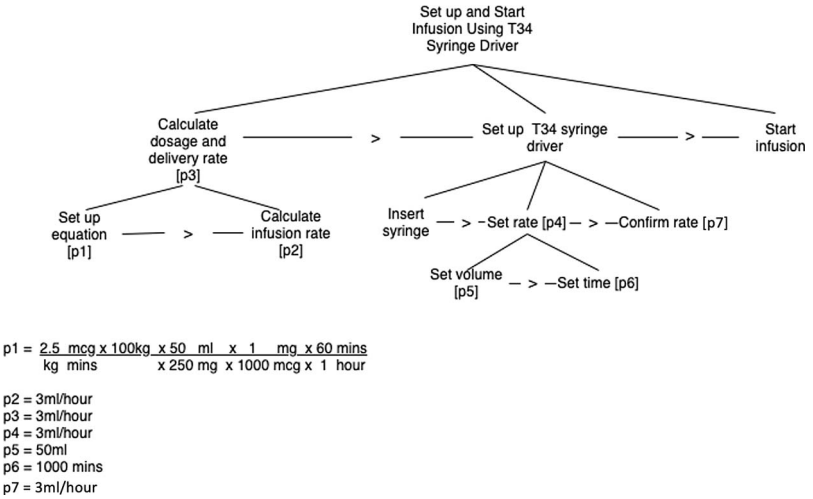


**FIGURE 15.** Combined parameterized model for dosage calculation and set up of T34 syringe driver.

that a user may have to deal with in real-world scenarios. By creating models of actual devices and comparing with the simulations, we can identify where the gaps are, which enables the educational tools to be extended to remove these gaps. Ideally, the numeracy tools should enable different types of number entry and device set up routines in order to fully prepare the nurses with required technical competence.

## 6. DISCUSSION

Our aim is to use task models as a common language to describe user goals, knowledge requirements and interactions with medical devices. By representing each of these as a task model we are able to combine and compare models in the manner described above to provide a number of benefits. Because the models are intended to be used in a 'lightweight' manner, we do not try to capture all aspects within the task models, but rather focus on the specific interaction parts that are common across the three domains (user tasks, device models and numeracy education tasks). While the use of task modelling in interactive system design and as part of a UCD approach is not new, we believe that our approach enables increased benefits from their use. Of course, in development approaches where full systems are described and developed within a task modelling environment (for example as seen in the Hamsters approach Barboni *et al.*, 2010), there are different benefits. A fully cohesive model expressed in a single language and tool enables a complete formal analysis, and potentially development of tests, from within a single environment. In addition, the structuring mechanisms introduced in [Forbrig *et al.*, 2014] allow for composition and re-use of sub-models and generic components (larger pieces of sub-task models). Our approach aims at similarly modular approach, but also one where we make use of different tools (at varying levels of abstraction and formality) depending on the particular development requirement throughout the process. Such an approach relies on the ability to combine models and move between notations with guarantees that we can preserve the properties from one set of models to another. While we have shown this to be true in earlier parts of our work we must also

demonstrate that this is true of the combinations of smaller task models and comparisons we have described.

It might be argued that without a single, complete model of all parts of the system we may not identify every possible mismatch or potential error that could occur. While we agree that this is the case we also believe that having smaller, more tractable models means that they are easier to use and include in a wide variety of existing development processes, which may ultimately be more beneficial. If we can provide an agnostic approach where small task models can be included as part of a model-based design process with the described benefits, this may ultimately be more useful than relying on a wholesale reliance on a large monolithic development process where models have the potential to become large and hard to understand (or build correctly).

The development of task models from the numeracy education examples, which represent authentic tasks, allows us to consider user knowledge as part of the task modelling an comparison process. Rather than describing explicit knowledge components and including these in standard task models (in the manner of Martinie *et al.*, 2013), we suggest that explicitly describing these as a task model makes it clearer when mismatches in understanding and actual device use are present. In this way, the education tasks represent the 'best practice' approach to solving a particular problem where we assume that if a user is sufficiently competent they will behave in this way. There is, of course, still potential for error if the user deviates from these steps, and inclusion of these types of slips and mistakes are not included in our approach.

The task model notation we have used in this paper is deliberately simple, it is not our intention to add to the already existing notations or tools. However, there may be benefits from integrating our work with such existing languages, particularly if we wanted to consider the integration of knowledge objects with task models in the manner proposed by Martinie et al. in [Martinie *et al.*, 2013]. There the authors describe how they might represent different types of knowledge of the user and how this relates to information about the system and what is required to complete sub-steps in the model. In a future work, it would be interesting to explore this approach further with the types of knowledge we have described here.

Bringing together the two domains in the manner described has the potential, therefore, to provide benefits to both. The formal models and device models can suggest areas that should be included in the technical competence and numeracy education. Likewise, the task models of the authentic numeracy tasks can indicate improvements or enhancements of the medical devices. Although we have introduced the idea of using such models to support users switching between different types of device (which may have subtle differences in how they are used), we have not elaborated on this here. Similarly we have not discussed how the combined models may be used to consider the use of multiple devices for a single patient. We leave these matters for future work; however, from these initial examples, we believe we have demonstrated the applicability of our methods in this area. Similarly the approach described is not restricted to the medical domain. There are many safety critical domains where there is a similar overlap between user training requirements, system use and system design where it would be beneficial to consider a similar approach.

## 7. CONCLUSIONS AND FUTURE WORK

In this paper we have laid the groundwork for a closer integration between software and device models used to improve design and use of safety critical medical systems with numeracy education in and for the clinical context.

We have shown how task models of user goals can be used as inputs to formal models of interactive systems, such as medical devices. We have also shown how task models can be derived from such formal models as well as specialized device task models. Task models used in this way can be used to improve design and identify and mitigate the effects of potential user errors. We also create task models from numeracy education examples, which describe knowledge-based tasks for medication delivery to ensure nursing staff have the necessary skills to perform medication dosage calculations and administer medication. The three types of task models can then be used in conjunction with each other and specialized and parameterized to inform user education and identify potential user errors.

Our main contributions here are demonstrating that by expressing properties of two different domains (interactive system modelling and numeracy education) in a common language—task models, we are able to compare and integrate models. In order to extend this work further we can now begin to consider deriving algorithms for traversing the task models in order to automate the process of combining and comparing the models. This may also require some ontological mapping to resolve naming differences and automate the understanding needed to identify types—such as units of time and measurement, etc. We also wish to investigate further how different types of number entry (five key interfaces vs. numeric keypads, for example) may lend themselves to calculation competence better than others and whether this can be identified from the task model comparisons. Such future work will allow us to explore the approach we have introduced here and investigate other ways of incorporating partial and lightweight models and task models within a larger development process.

## REFERENCES

Annett, J. and Duncan, K. (1967) *Task Analysis and Training Design*, 22 pages. ERIC Clearinghouse, Washington, DC.

Barboni, E., Ladry, J., Navarre, D., Palanque, P. A. and Winckler, M. (2010) Beyond modelling: an integrated environment supporting co-execution of tasks and systems models. In *EICS*, pp. 165–174. ACM.

Berndt, E., Furniss, D. and Blandford, A. (2015) Learning contextual inquiry and distributed cognition: a case study on technology use in anaesthesia. *Cognit. Technol. Work*, 17, 431–449.

Blandford, A., Buchanan, G., Curzon, P., Furniss, D. and Thimbleby, H. (2010) Who's looking? Invisible problems with interactive medical devices. In *Proc. First Int. Workshop on Interactive Systems in Healthcare*, pp. 9–12. ACM Special Interest Group on Computer–Human Interaction, USA.

Blandford, A., Butterworth, R. and Good, J. (1997) Users as rational interacting agents: formalising assumptions about cognition and interaction. In *DSV-IS*, pp. 45–60. Springer.

Blandford, A. and Furniss, D. (2005). Dicot: a methodology for applying distributed cognition to the design of teamworking systems. In *DSV-IS*, Vol. 3941 of Lecture Notes in Computer Science, pp. 26–38. Springer.

Blandford, A., Pietro, G. D., Gallo, L., Gimblett, A., Oladimeji, P. and Thimbleby, H. W. (2011) Engineering interactive computer systems for medicine and healthcare (EICS4Med). In *EICS*, pp. 341–342.

Bodkin, H. (2021) Hunt orders probe into faulty opioid syringe pumps amid allegations thousands may have died. *The Telegraph*.

Bowen, J. (2015) Creating models of interactive systems with the support of lightweight reverse-engineering tools. In *Proc. 7th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS 2015*, Duisburg, Germany, June 23–26, 2015, pp. 110–119.

Bowen, J. and Reeves, S. (2006) Formal models for informal GUI designs. In *1st Int. Workshop on Formal Methods for Interactive Systems, Macau SAR China, 31 October 2006*, Vol. 183, pp. 57–72. Electronic Notes in Theoretical Computer Science. Elsevier.

Bowen, J. and Reeves, S. (2007). Refinement for user interface designs. In Curzon, P. and Cerone, A. (eds) *Proc. 2nd Int. Workshop on Formal Methods for Interactive Systems (FMIS 2007)*, Vol. 208, pp. 5–22. Electronic Notes in Theoretical Computer Science. Elsevier, Lancaster University, UK.

Bowen, J. and Reeves, S. (2008) Formal models for user interface design artefacts. *Innov. Syst. Soft. Eng.*, 4, 125–141.

Bowen, J. and Reeves, S. (2013) Modelling safety properties of interactive medical systems. In *Proc. 5th ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS '13*, pp. 91–100. ACM, New York, NY, USA.

Bowen, J. and Reeves, S. (2017) Combining models for interactive system modelling. In *Handbook of Formal Methods in Human–Computer Interaction*, pp. 161–182. Springer International Publishing.

Calvary, G., Coutaz, J. and Thevenin, D. (2001) Supporting context changes for plastic user interfaces: a process and a mechanism. In Blandford, A., Vanderdonckt, J., Gray, P. (eds) *Joint Proceedings of HCI'2001 and IHM'2001*, pp. 349–363. Springer.

Campos, J. and Harrison, M. (2011) Modelling and analysing the interactive behaviour of an infusion pump. *ECEASST*, 11.

Card, S., Moran, T. and Newell, A. (1983) *The Psychology of Human Computer Interaction*. Lawrence Erlbaum Associates.

Card, S. K., Moran, T. P. and Newell, A. (1980) The keystroke-level model for user performance time with interactive systems. *Commun. ACM*, 23, 396–410.

Coben, D. and Bowen, J. (2019) Safety first: combining task models of medical devices with numeracy skills and technical competence. *Adults Learn. Math.* 14, 39–60.

Coben, D. and Weeks, K. (2014) Meeting the mathematical demands of the safety-critical workplace: medication dosage calculation problem-solving for nursing. *Educational Studies in Mathematics. Special Issue on Vocational Education and Workplace Training*, Vol. 86, 2, pp. 253–270.

Dittmar, A. and Forbrig, P. (2003) Higher-order task models. In *DSV-IS*, Vol. 2844 of Lecture Notes in Computer Science, pp. 187–202. Springer.

European Education and Culture Commission (2008) *The European Qualifications Framework for Lifelong Learning (EQF)*. Office for Official Publications of the European Communities, Luxembourg. ISBN 978-92-79-08474-4.

FDA (2010) U.S. Department of Health and Human Services. Examples of reported infusion pump problems. http://www.fda.gov/medicaldevices/productsandmedicalprocedures/generalhospitaldevicesandsupplies/infusionpumps/ucm202496.htm (accessed April 21, 2021).

Forbrig, P., Martinie, C., Palanque, P., Winckler, M. and Fahssi, R. (2014) Rapid task-models development using sub-models, sub-routines and generic components. In Sauer, S., Bogdan, C., Forbrig, P., Bernhaupt, R., Winckler, M. (eds) *Human-Centered Software Engineering*, pp. 144–163. Springer Berlin Heidelberg, Berlin, Heidelberg.

Galbraith, P. L., Henn, H.-W. and Niss, M. (2007) *Modelling and Applications in Mathematics Education: The 14th ICMI Study*. New ICMI Study Series. Springer US.

Harrison, M. D., Masci, P., Campos, J. C. and Curzon, P. (2017) Verification of user interface software: the example of use-related safety requirements and programmable medical devices. *IEEE Trans. Hum.Mach. Syst.*, 47, 834–846.

Johnson, C. W. (2011) *An Introduction to Human Error, Interaction and the Development of Safety-Critical Systems*. Ashgate, Farnham, Surrey, UK. ISBN 978-0-7546-7580-8k.

Kieran, C., Doorman, M. and Ohtani, M. (2015) Frameworks and principles for task design. In Watson, A., Ohtani, M. (eds) *Task Design In Mathematics Education: An ICMI Study 22*, pp. 19–81. Springer International Publishing, Cham.

Kohn, L. T., Corrigan, J. M. and Donaldson, M. S. (2000) *To Err Is Human: Building a Safer Health System*. The National Academies Press, Washington, DC.

Leufer, T. and Holdforth-Cleary, J. (2011) Medication management—last line of defence. *World Irish Nurs.*, 19, 48–50.

Martinie, C., Palanque, P. A., Navarre, D., Winckler, M. and Poupart, E. (2011) Model-based training: an approach supporting operability of critical interactive systems. In *EICS*, pp. 53–62. ACM.

Martinie, C., Palanque, P. A., Ragosta, M. and Fahssi, R. (2013) Extending procedural task models by systematic explicit integration of objects, knowledge and information. In *European Conference on Cognitive Ergonomics 2013, ECCE '13*, Toulouse, France, August 26–28, 2013, pp. 23:1–23:10.

Mori, G., Paternò, F. and Santoro, C. (2002) CTTE: support for developing and analyzing task models for interactive system design. *IEEE Trans. Softw. Eng.*, 28, 797–813.

Palanque, P. A., Ladry, J., Navarre, D. and Barboni, E. (2009) High-fidelity prototyping of interactive systems can be formal too. In *HCI (1)*, Vol. 5610 of Lecture Notes in Computer Science, pp. 667–676. Springer.

Palm, T. (2009). *Theory of Authentic Task Situations*, pp. 3–19. Sense Publishers, Rotterdam.

Paterno, F. (2001) *Task Models in Interactive Software Systems*, pp. 817–836. World Scientific Publishing Company.

Paternò, F., Mancini, C. and Meniconi, S. (1997) Concurtasktrees: a diagrammatic notation for specifying task models. In *Human–Computer Interaction, INTERACT '97, IFIP TC13 International Conference on Human–Computer Interaction*, 14th–18th July 1997, Sydney, Australia, pp. 362–369.

Rajkomar, A. and Blandford, A. (2012) Understanding infusion administration in the ICU through distributed cognition. *J. Biomed. Inform.*, 45, 580–590.

Ruksenas, R., Back, J., Curzon, P. and Blandford, A. (2008) Formal modelling of salience and cognitive load. *Electr. Notes Theor. Comput. Sci.*, 208, 57–75.

Seffah, A., Vanderdonckt, J. and Desmarais, M. C. (2009) *Human-Centered Software Engineering: Software Engineering Models, Patterns and Architectures for HCI*. Springer Publishing Company, Incorporated.

Shepherd, A. (1989) Analysis and training in information and technology tasks. In Diaper, D. (ed.) *Task Analysis for Human–Computer Interaction*, Chapter 1, pp. 15–55. Ellis Horwood.

Thimbleby, H. (2000) Calculators are needlessly bad. *Int. J. Hum. Comput. Stud.*, 52, 1031–1069.

Thimbleby, H. (2015) Safer user interfaces: a case study in improving number entry. *IEEE Trans. Softw. Eng.*, 41, 711–729.

Turner, J., Bowen, J. and Reeves, S. (2017) Supporting interactive system testing with interaction sequences. In *Proc. ACM SIGCHI Symposium on Engineering Interactive Computing Systems, EICS 2017*, Lisbon, Portugal, June 26–29, 2017, pp. 129–132.

Vipond, S. (2016) Pioneering medical device training in the digital age. *Learning Solutions Magazine*, 2027.

Watson, A. and Ohtani, M. (2015) *Task Design in Mathematics Education, an ICMI study 22*. Springer.

Weeks, K., Hutton, M., Young, S., Coben, D., Clochesy, J. and Pontin, D. (2013a) Competency modelling and diagnostic error assessment in medication dosage calculation problem-solving. *Nurse Education in Practice*, Safety in numbers, 2, Article 13.

Weeks, K. W., Hutton, M. B., Coben, D., Clochesy, J. M. and Pontin, D. (2013b) Safety in numbers 3: authenticity, building knowledge & skills and competency development & assessment: the abc of safe medication dosage calculation problem-solving pedagogy. *Nurse Education in Practice*, 13, e33–e42.