EXTENDING STPA TO IMPROVE THE ANALYSIS OF USER INTERFACE SOFTWARE IN MEDICAL DEVICES

Yi Zhang, Ph.D.
Center for Devices and Radiological Health (CDRH)
U.S. Food and Drug Administration (FDA)
YI.ZHANG2@FDA.HHS.GOV

Based on joint work with Paolo Masci and Jose Campos (University of Minho, Portugal) and Paul Jones (FDA/CDRH)
CONTRIBUTIONS

• Extending the STPA analysis to:
  – Identify design issues in medical device user interface (software) that likely enable or promote use errors
  – Define formal requirements for mitigating the identified design issues

• A case study on a medical device to evaluate the effectiveness of the extension
OUTLINE

• Medical device use errors and user interface software
• Extending STPA to improve analysis of user interface software
• Case study
MEDICAL DEVICE AND SOFTWARE

• U.S. Food and Drug Administration is charged with ensuring the safety and effectiveness of medical devices before they enter into the market.
  ▪ Medical devices are classified as Class I, II, and III based on intended use and indications for use.
  ▪ Class II devices need premarket clearance and Class III devices need premarket approval, but exceptions exist.

• Software in medical devices receives regulatory oversight (pre-market review, quality system regulations, and post-market reporting).
  ▪ Level of Concern (LoC): the severity of harm potentially caused by software risks before mitigation.
  ▪ Different LoCs means device makers need to submit different levels of software information.
  ▪ Software in a medical device (SiMD) vs. Software as a medical device (SaMD).
• **Use Error:**
  User action or lack of user action while using the medical device that leads to a different result than that intended by the manufacturer or expected by the user. (IEC 62366)

• **Use Error or User Error?**

  [Use Error] was chosen over ... “human error” because not all errors associated with the use of medical device are the result of oversight or carelessness of the part of the user of the medical device. Much more commonly, use errors are the direct result of poor user interface design.” (IEC 62366)
An Example Medical Device Use Error

**USE Error**
Surgeon deviates from the standard procedure and presses the button to retract the catheter when it and the guide wire is being installed.

**Result**
Unexpected retraction of the catheter, even after the retract button is released.

**Health Risks**
Partial dissection, perforation, hematoma, …

Robotic Surgical System Recalled in 2015
(https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfRES/res.cfm?id=135134)
Use errors can occur due to:

- Limitations and variability of the user’s ability to operate the device.
- Unsuitable environmental conditions (lighting, noise, vibration, etc.).
- Poor user interface (UI) design.
  - Size and shape
  - Input widgets (buttons, switches, knobs)
  - Output widgets (display, light, LED, etc.)
  - Connections and accessories
  - Packaging and labeling (including manuals)
  - Software

Human Factors (HF) Considerations of Medical Device Use Safety
Adapted from FDA Guidance on Applying HF and Usability Engineering to Medical Devices, available at
https://www.fda.gov/downloads/MedicalDevices/.../UCM259760.pdf
**AN EXAMPLE MEDICAL DEVICE USE ERROR (CONT’D)**

**USE Error**
Surgeon deviates from the manual and presses the button to retract the catheter when it and the guide wire is being installed.

**Results**
Unexpected retraction of the catheter, even after the retract button is released, causing risks like hematoma or partial dissection.

**Root Cause**
The UI software failed to prevent inappropriate press of the retract button and then to recognize its release.

Robotic Surgical System Recalled in 2015
(https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfRES/res.cfm?id=135134)
**An Other Example**

Use Error

Nurse incorrectly entered the infusion rate as 1105 ml/hr, 10 times more than the prescribed rate 110.5 ml/hr.

Root Cause

The UI software mistakenly discards the decimal point of the entered infusion rate between 100.1 to 1200 ml/hr without notifying the user [1].

WHAT IS UI SOFTWARE?

I/O Devices

I/O Device Driver

Input Interpreter

Output Render

Interaction Logic

User Interface Software

Translate use actions to input signals; Instruct output devices to present user feedbacks

Translate user input signals to commands

Determine where and how to present user feedbacks

• Process and respond to user commands;
• Determine the content and timing of user feedback;
• Interact with the rest of device software

Adapted from A Generic User Interface Architecture for Analyzing Use Hazards in Infusion Pump Software, Masci et al., 5th Workshop on Medical Cyber Physical Systems, pp. 1-14, 2014.
HOW COMMON ARE UI SOFTWARE DEFECTS?

According to our study on FDA medical device recall data from 09.2012 to 08.2015

<table>
<thead>
<tr>
<th></th>
<th>Year I</th>
<th>Year II</th>
<th>Year III</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device Recalls</td>
<td>2294</td>
<td>2753</td>
<td>2724</td>
<td>7771</td>
</tr>
<tr>
<td>Recalls caused by software defects</td>
<td>306</td>
<td>323</td>
<td>284</td>
<td>913</td>
</tr>
<tr>
<td>Recalls caused by UI software defects</td>
<td>148</td>
<td>145</td>
<td>128</td>
<td>421</td>
</tr>
</tbody>
</table>

Recalls due to UI software defects account for 46.11% of recalls caused by software defects

The majority of UI software defects are (subtle) interaction logic flaws
Gaps and Challenges

• UI software defects constitute an important cause of medical device user errors.

• Device manufacturers and other stakeholders often address the safety of UI from HF (e.g., ergonomics) perspective.

• Software engineers typically are not involved early on in the UI design.
  • Incomplete analysis of software-related causes to use errors.
  • Inadequate design and verification of software measures for mitigating use-related hazards.
• Medical device use errors and user interface software
• Extending STPA to improve analysis of user interface software
• Case study
Risk Management Process for Medical Devices

Risk Analysis
- Identification of Critical User Tasks and UI Components Involved
- Evaluation of Use-Related Hazards

Risk Control (design, implement, and test)
- HF Validation Testing
- Design & Implement Control Measures
- Evaluation of Residual Risks

Evaluated Overall Residual Risks

Risk Management Process Defined in ISO 14971

Risk Management Process for Use-Related Hazards Recommended by FDA’s HF Guidance
IDENTIFICATION OF USE-RELATED HAZARDS

• Establishes the foundation for subsequent risk management activities completeness vs. reasonableness

• Approaches recommended in FDA HF guidance:
  § Bottom-Up Analysis: Failure Mode Effect Analysis (FMEA)
  § Top-Down Analysis: Fault Tree Analysis (FTA)
  § Known use-related issues

• Limitations:
  § Device-centric: need to bridge device design issues and their impact to users
  § Limited guidance is provided to developers in identifying UI (software) design issues
STPA FOR IDENTIFYING USE-RELATED HAZARDS

Unsafe Control Actions (UCAs):
- Control action not performed or followed;
- Unexpected control action performed;
- Control action performed at the wrong time or in a wrong order;
- Control action stops too soon or lasts too long.

Limited guide words are provided for exploring three aspects of UI design issues enabling UCAs: feedback, mental model, and external information.
Detailed guidelines are needed to support systematic identification of UI design issues as causal factors to human UCAs.

OUR SOLUTION: HUMAN-STPA

Rasmussen’s Step-Ladder Model to support the analysis of human cognitive causes of UCAs.

Human Cognitive Model

STPA

Human-STPA

Rasmussen’s Decision Ladder Model

HF & Usability Standards

Causal factors categories are derived from UI design principles defined in HF & usability standards.
HOW HUMAN-STPA IMPROVES STPA?

- Introduced 16 new causal factors categories
  - Based on design principles from IEC 62366 and AMMI HE75 standards
  - Provide developers detailed guidelines in identifying use-related safety issues in medical device UI design

- Improved methods for defining safety requirements:
  - Formalizes safety requirements as theorems that can proved against UI design
  - Executable UI safety reference model demonstrates the intended meaning of safety requirements

**Human-STPA Process**

**Step 1**
- Define system boundaries
- Identify human UCAs

**Step 2**
- Identify UI components necessary for UCAs
- Apply causal factors categories to each UI component to identify issues

**Step 3**
- Define safety requirements to mitigate the identified UI issues
### Causal Factor Categories

<table>
<thead>
<tr>
<th>Original STPA Categories</th>
<th>New Causal Factors Categories</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feedback</td>
<td>Consistency of feedback, Complexity of feedback</td>
</tr>
<tr>
<td></td>
<td>Availability of feedback, Salience of feedback</td>
</tr>
<tr>
<td>Mental model</td>
<td>Reversibility of control actions</td>
</tr>
<tr>
<td></td>
<td>Responsiveness to control actions</td>
</tr>
<tr>
<td></td>
<td>Predictability of controls</td>
</tr>
<tr>
<td></td>
<td>Forgiveness for erroneous control actions</td>
</tr>
<tr>
<td></td>
<td>Availability of controls</td>
</tr>
<tr>
<td></td>
<td>Affordance of controls</td>
</tr>
<tr>
<td></td>
<td>Consistency of controls</td>
</tr>
<tr>
<td></td>
<td>Consistency with clinical workflows</td>
</tr>
<tr>
<td></td>
<td>Complexity of controls</td>
</tr>
<tr>
<td>External information</td>
<td>Availability of user manuals</td>
</tr>
<tr>
<td></td>
<td>Consistency with user manuals</td>
</tr>
<tr>
<td></td>
<td>Complexity of user manuals</td>
</tr>
</tbody>
</table>

Full details of the causal factors categories are available in [A Hazard Analysis Method for Systematic Identification of Safety Requirements for User Interface Software in Medical Devices](https://example.com), Masci et al., SEFM’17, pp. 284-299, 2017.
Robotic Surgical System Recalled in 2015
(https://www.accessdata.fda.gov/scripts/cdrh/cfdocs/cfRES/res.cfm?id=135134)

**UCA: Retract the catheter too soon**
UI component involved: catheter retract button

**Category: Forgiveness of erroneous control actions**
Safety interlocks are not available to prevent accidental activation of critical control actions performed by the user, or block/mitigate foreseeable user operation errors.

**UI Design Issue:**
The UI software does not provide means to prevent the press of the catheter retract button when it is not safe to do so.
Note: other UI design issues might also exist.
Diabetes Management App silently resets bolus dose recommendation when the user changes phone orientation

**Example Causal Factors Category (cont’d)**

**UCA:** Change phone orientation while configuring bolus dosage
**UI component involved:** phone

**Category:** Predictability of controls
*The UI does not provide means to help the user anticipate the effects or the consequences of control actions.*

**UI Design Issue:** Lack of feedback
*The UI software does not provide appropriate feedback when the user performs the UCA and the bolus dosage is reset.*
*Note: other UI design issues might also exist.*
• Medical device use errors and user interface software
• Extending STPA to improve analysis of user interface software
• Case study
**CASE STUDY: GANTRY-2 SYSTEM**

- Experimental radiation therapy machine
- Two control consoles: central and remote
- Human operator tasks:
  - Set up system/load treatment plan/start treatment
  - Monitor patient and system
- User manual and UI design documentation are available
- Classic STPA has been applied to the UI design of Gantry-2 System [1]

We analyzed the Gantry-2 System using the same control structure and documentation as used in the original analysis [1].

- Identified all problems reported in the original analysis, and
- several new issues overlooked in the original analysis

For UCA “Treatment start command is activated when there is no patient to be treated”, Human-STPA reported 7 new UI software issues not reported in [1].
### Example New UI Issue Identified

<table>
<thead>
<tr>
<th>UI Design Issue</th>
<th>UI software does not provide feedback when patient is not ready</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI Component(s)</td>
<td>Display</td>
</tr>
<tr>
<td>Causal Factors</td>
<td>Availability of feedback: Feedback reporting important</td>
</tr>
<tr>
<td>Category</td>
<td>information or events is erroneous, not visible, partially</td>
</tr>
<tr>
<td></td>
<td>visible, or is visible at the wrong time.</td>
</tr>
<tr>
<td>Safety Requirement</td>
<td>“Patient not ready” alerts shall be displayed on the UI.</td>
</tr>
<tr>
<td>Formalized Safety Requirement</td>
<td>FORALL (pre, post: State, c: Console): (init(pre)→ pt_status_visible (on(c)(pre))) ^ (pt_status_visible(pre) ^trans(pre , post)) → pt_status_visible(post))</td>
</tr>
</tbody>
</table>

For every system state `pre`, patient status is visible (`pt_status_visible`) on any control console `c`, and this is also true for any subsequent system state of `pre`.
### Example New UI Issue Identified (Cont’d)

<table>
<thead>
<tr>
<th>UI Design Issue</th>
<th>UI software fails to block the operator’s accidental press on the start command (e.g., during system maintenance).</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI Component(s)</td>
<td>Start command</td>
</tr>
<tr>
<td>Causal Factors Category</td>
<td>Forgiveness for erroneous control actions</td>
</tr>
<tr>
<td>Safety Requirement</td>
<td>The UI software shall not accept a start command when the patient is not ready. For every system state ( st ), if patient status is not ready, then the Start Command ( \text{per_start}(st) ) is not allowed.</td>
</tr>
<tr>
<td>Formalized Safety Requirement</td>
<td>\text{FORALL} (st: State): \text{controller}(st) \wedge \text{patient} = \text{NOT_READY} \rightarrow \text{NOT per_start}(st)</td>
</tr>
</tbody>
</table>
**EXAMPLE NEW UI ISSUE IDENTIFIED (CONT’D)**

<table>
<thead>
<tr>
<th>UI Design Issue</th>
<th>If the treatment activation sequence is interrupted, the UI software always resumes the sequence from where it was interrupted, and does not provide the operator means to stop/abort/restart the activation sequence.</th>
</tr>
</thead>
<tbody>
<tr>
<td>UI Component(s)</td>
<td>UI commands involved in the activation sequence.</td>
</tr>
<tr>
<td>Causal Factors</td>
<td>Reversibility of control actions: Functions for reversing the effect of control actions are not available.</td>
</tr>
<tr>
<td>Category</td>
<td></td>
</tr>
<tr>
<td>Safety Requirement</td>
<td>A control to cancel the treatment activation sequence shall be available at all UI screens for both control consoles.</td>
</tr>
<tr>
<td>Formalized Safety Requirement</td>
<td>cancel_activation_th: FORALL (st: State , c: Console) controller(st).mode /= off → controller(stop(c) (st)).mode = ready</td>
</tr>
</tbody>
</table>
Formal safety requirements enables the development of UI safety reference models.

• The models’ compliance to intended use safety can be established through formal verification (e.g., theorem proving).

• Safety reference models can be used to challenge the actual UI software design (e.g., using model-based test generation).

PVS Model for Gantry-2 System
The full PVS model and analysis results are available at https://goo.gl/7ftTv.
DISCUSSIONS

• Formal verification vs. other evaluation methods (e.g., expert review)
  
e.g., Consistency with user manuals and Complexity of feedback

• Identification of human UCAs vs. Identification of critical tasks.
  
  Analytical approaches for identifying critical user tasks, e.g. contextual inquiry or formative evaluations, can help the identification of UCAs.

• Need enhancement on identifying UI issues with supporting similar control actions.
FUTURE WORK

• More comprehensive evaluation on realistic systems.

• Automation of applying causal factor categories to UI design.

• Establishing UI safety reference models for classes of devices
Questions?