December 2021

DETECTING ANOMALOUS USER RESPONSES TO HAPPY-PATH FLOW

Rob Hanton

DETECTING ANOMALOUS USER RESPONSES TO HAPPY-PATH FLOW

AUTHORS:
Rob Hanton

ABSTRACT

Presented herein are techniques through which video and other potential inputs can be used by a software product or system in order to detect sudden increases in user frustration or confusion during certain system transitions, such as a call being hung up manually, in order to automatically detect when such things occur in a way that is unexpected by the user. For such increases, additional information and/or interactions can be provided to the user in accordance with techniques herein to help the user better understand and/or remediate the issue, such as providing a dialog box that explains that the other participant hung up the call manually, and providing a button to immediately call them back.

DETAILED DESCRIPTION

In recent years, a major focus for providing a good user experience (UX) design has been to provide a smooth experience during usage that minimizes the number of actions a user must take to achieve a desired outcome. Even when a range of configuration options and notifications are available to a user, the typical user is now generally only exposed if the user deviates, by choice or not, from the default, "happy-path" experience. In one example, this might mean clicking an 'advanced' menu to set their own bandwidth limits. In another example, this might be a notification about a call state that might only appear if a video call were to end due to unexpected circumstances (e.g., a network connectivity issue).

For the most part this style of design is effective, as it allows a user to use the software quickly and seamlessly with a minimum of training, while often minimizing notification fatigue. However, the specific choice to suppress notifications can be problematic in cases where, even when the system is following a standard flow, it may have changed state in a way that the user was not expecting. In such cases, the lack of notifications can add to a user's confusion and, given the need to not normally look at advanced options and notifications, a user may not be able to discover what happened even if the information is available.

Consider an example involving making a 1:1 call when one side, say, user Alice, ends a call with another side, say user Bob. Normally it is mutually understood by both parties that the call has come to an end. In this example, modern systems generally will not provide an explicit notification to either side that user Alice ended the call manually. However, in cases where user Alice ended the call accidentally, user Bob and even user Alice may be confused as to why the call has ended given the lack of feedback.

In order to address such issues, this proposal involves techniques that provide for the ability to detect rapid increases in user frustration/annoyance/confusion at moments of major transitions in the 'happy-flow' of a software product that have the scope for being triggered accidentally or unexpectedly by participant action, and, in those circumstances, to present the user with additional information and potentially actions they can take to both inform them of what has happened and aid them in resolving their confusion. Consider various example scenarios involving users Alice and Bob, as shown below in Figure 1, which will be discussed in further detail herein in relation to various features of the techniques that may be provided by this proposal. The various example scenarios shown in Figure 1 illustrate estimates of Bob's confusion levels over time and, in particular, with relation to when a call between Alice and Bob ends.

The first example scenario (Example 1), illustrates the baseline scenario, noted above, in which it is assumed that Alice ends the call at the end of the conversion in a way that is mutually understood and, as a result, Bob's estimated confusion levels remain unchanged. In this scenario, no significant spike in confusion would be detected for Bob, so a system configured in accordance with this proposal would take no extra actions.

In another example scenario, such as shown in Example 2, consider that Alice and Bob are involved in an angry, emotional conversation in which Bob's level of frustration may already be high. Thus, in the example, if Alice abruptly hangs up, no significant change in frustration may be detected for Bob, so the system would take no extra actions in accordance with this proposal.

Example 1

- Alice calls Bob
- Alice and Bob have a normal conversation
- Alice hangs up at the end of the conversation as Bob is expecting

No significant spike in confusion estimation recorded on hang-up, so the system takes no extra actions.

Example 2

- Alice calls Bob
- Alice and Bob have an angry, emotional conversation
- Alice hangs up at the end of the conversation as Bob is expecting

Estimated confusion levels are already high on hangup and hence no spike is recorded, so the system takes no extra actions

Example 3

- Alice calls Bob
- Alice and Bob have a normal conversation
- A network event causes Alice's call to terminate unexpectedly

Estimated confusion levels spike when Alice's call terminates unexpectedly, but because this is not a 'happy path' transition the system will already display information regarding what occurred and no extra actions related to confusion estimates are taken

Example 4

- Alice calls Bob
- Alice and Bob have a normal conversation
- Alice accidentally hits the 'hang up' button while trying to share her screen

A significant spike in user confusion is estimated during a manual happy-path transition, and hence the system provides additional information and actions to Bob

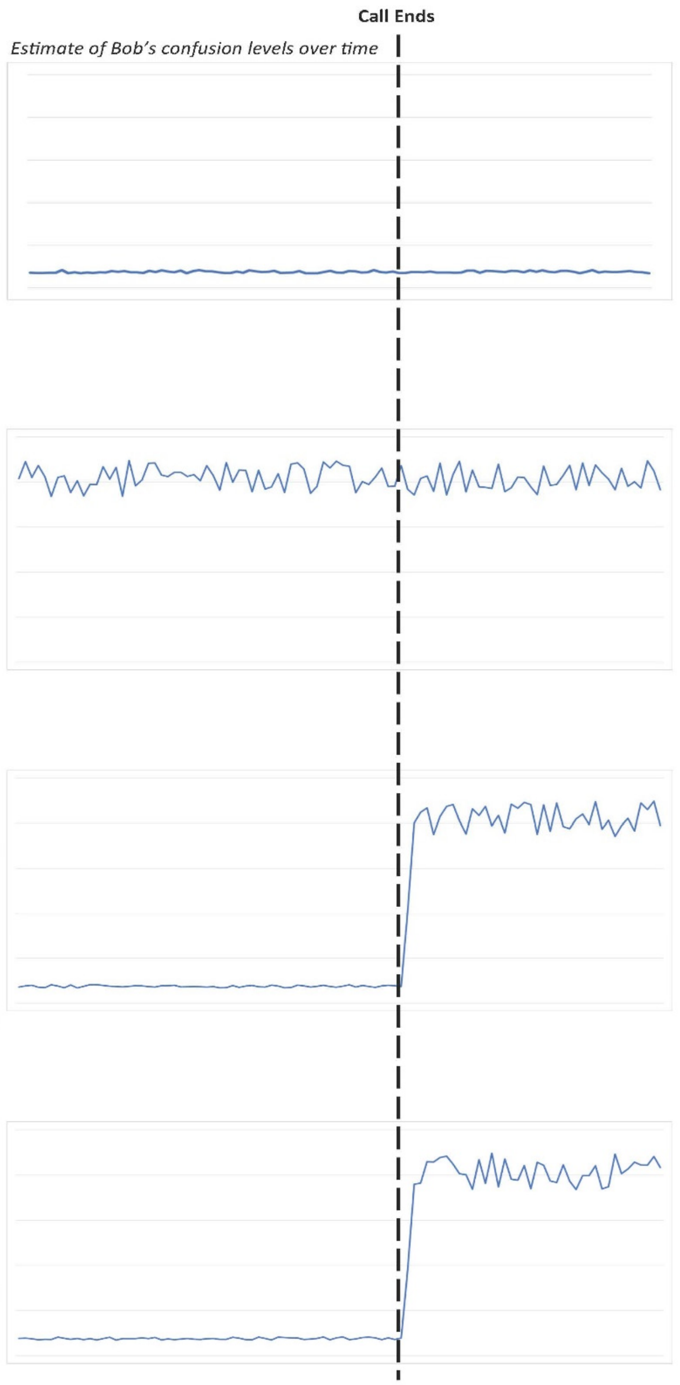

Figure 1: Example Scenarios

Example 3 illustrates a case where a network event causes Alice's call to terminate unexpectedly. Bob's confusion level rises quickly and rapidly, but in this case the system is already aware that this is no a "happy path" transition and so will already display a notification that the transition was caused by a network issue. This does not require any monitoring of Bob's confusion levels.

However, consider another example scenario, such as shown in Example 4, in which Alice and Bob are in a video call, but Alice accidentally hangs the call up while trying to change share her screen, which, as shown in Example 4, leaves Bob confused at why the call has suddenly ended. In this example scenario, the system can detect that Bob is showing a major and rapid increase in confusion as he was not expecting the call to end, and can provide a notification with additional information, such as notification saying "This call was hung up by Alice at 09:23:15" and may include a 'redial Alice' button alongside the normal 'close dialog' button. Such a solution automatically lets Bob know what happens, and gives him a rapid way to remediate the issue with a single action presented very clearly.

Detecting rapid changes in user emotion in the case of audio/video conferencing can be provided via audio/video that the system is already in a position to capture as part of the conferencing functionality. In cases outside of calls such as monitoring emotions on call termination, this may involve continuing to run the audio/video capture for a few seconds subsequent to the call ending. In some instance, such as for privacy reasons, it may be desirable to provide the ability for the user to opt in/out of such functionality.

Various mechanisms may be used to estimate user annoyance/frustration/confusion, such as, for example:

- Recognizing facial expression of a user's video;
- Performing sentiment analysis around the tone of a user's voice; and/or
- Detecting specific keywords and/or phrases in a user's audio relevant to the system transition (e.g., "Hello", "Are you there", etc.).

During operation, the mechanisms for detecting user annoyance, frustration, confusion, etc. should be run either immediately on transition or continually prior to the transition, as well as after transition, so that a change in a user's emotions immediately post-transition can be estimated. If multiple mechanisms are used, the confidence levels they use can be combined in order to produce a more robust estimate. It may be that some mechanisms may only be available under some circumstances. For instance, an implementation may use facial expression detection in all cases, but may also be able to combine it with the detection of specific keywords/phrases for English-language speakers to produce a more robust estimate.

In some instances, an implementation might alternatively (or additionally) implement an ML-based classifier to determine such moments. This could be trained by capturing one or more user inputs such as audio, video, mouse movements, etc. for a few seconds after the system transition in question and uploading these to a central repository. Given a sufficiently large body of such recordings they could then be manually classified as an expected or unexpected transition, and the data could then be used to train a binary classifier to provide a confidence estimate.

In some instances, when certain happy-path transitions occur in the software flow that do not usually provide any explicit notification (such as a call ending due to deliberate user action) but which can, in certain circumstances, be triggered unexpectedly to one or more users, the software product can calculate a confidence estimate that the transition was unexpected and confusing. If this value was above a certain threshold the software product may choose to take specific actions, such as showing a notification explaining precisely why the transition occurred (e.g., "User X hung up the call") and/or presenting additional options (such as a 'redial user X' button as part of the notification).

In some instances, a system might utilize multiple thresholds for detecting different levels of confusion or frustration, which might result in different actions at different levels of confidence. For instance, in the event that Alice ends a call and the confidence estimation for Bob is below a threshold for an explicit notification, but above a secondary, lower threshold, the system might instead show a ‘pulsing’ animation on a button that shows advanced diagnostics that will tell Bob why the call ended, in order to subtly draw attention to it.

This concept has the advantage that a video conferencing system can continue to offer a smooth user experience with a minimum of interruptions and notifications, while in the edge case of unexpected cases, such as call disconnections due to user error, these transitions can be detected as being unexpected and a user can be automatically presented with information to let them know what has happened, along with controls to remedy the issue. In the case of video conferencing, this can be provided by leveraging resources that are already available (audio/video of the user).

Unlike conventional systems that typically measure user frustration/confusion and use this measure to trigger UX changes, rather than continually monitoring and taking actions, techniques provide by this proposal focus on detecting a significant increase in a user's measured frustration/confusion during specific user-initiated transitions in a program, thereby providing for the ability to determine that the user was not expecting a given transition and intervening to provide additional information as to why it occurred and potentially providing additional controls to allow the user to take actions to address the issue.