Should components be able to receive a returning arrow?

Introduction

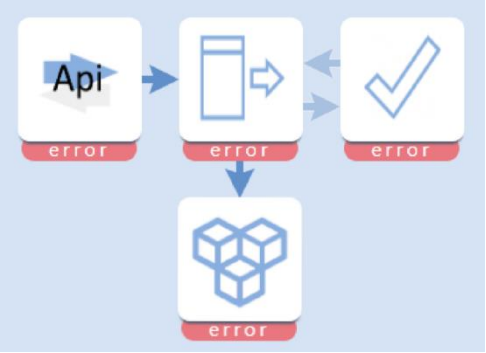
In this document I will be researching if components should have the function to receive a returning arrow. This will include research on what ways there should be an arrow needed that returns to the component before, research on how relevant these functionalities are and if these will be used often. I will also be researching if there are other functionalities that could replace an returning arrow to see if the arrows are even needed or if the other functionality is as reliable and user-friendly. I will also do a little study on the user friendliness of the functionality as we find user experience very important and it shouldn't obstruct this.

Methods I used for this research

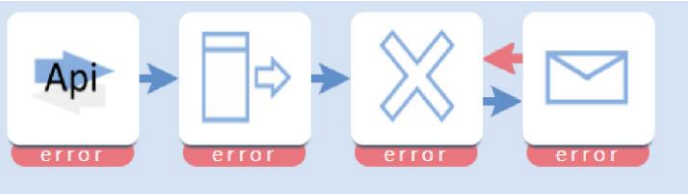
For this research I mostly used the library method, with literature study. As this document requires a lot of research based on functionalities and creations that already exists I found this the best way to find an answer for this research question. I also asked my stakeholders what they thought, by means of just sitting around the table and questioning them as their vision and opinion are important to me. I also used a research I have made earlier to see if the question we have to answer in this document corresponds with the UX, as this is also a very important topic.

Which functionalities in the app could need an returning arrow?

To know if it is relevant for us to make an returning arrow we first need to find what kind of functionalities could be relevant for this.
First component that could use an arrow that returns is a component that needs to check something, return an answer on that check and then return to the 'main' course it was going through. For example, a component receives the answer 10 and needs to check if this is higher than 12 to continue, this component gives the number 10 to a component to check if it is higher then 12 and returns that this is false and goes on with the main flow. It would look a little like this as shown down below;



Another example where a returning arrow could come in handy is when an error occurs that it should retry that component, or the component before that. When something fails in the flow it shouldn't immediately give an error, as sometimes something intern goes wrong and it could still continue when it retries. So an arrow that goes back to a component could be used, though it should have a maximum of tries that it can do as otherwise there is a possibility that you can end up in an endless loop that will continue until the end of time, that's why with a retry functionality it should have a max of tries.



Are there functionalities that could replace a returning arrow?

Are returning arrows relevant? Could they be replaced by other functionalities that work just as easy or even better? These are questions I need to ask myself doing this research as this can conclude if returning arrows are even needed.
Looking at the first picture and example given in the sub question before this paragraph. You can see there is an extension that returns something and goes on with the 'main' flow. What we could be experiencing when making it like this is recursion (source 8), as something is defined in terms of itself it could cause for a never ending loop also known as the 'halting problem' (source 9), this means that because of the component that returns it could return forever. A way this could be handled differently is by fusing the components that have an arrow in between, a functionality that ThingsDB would like to have is that you can make flows and make separate components from them, this means that you could make a flow like this;

And save them as a component and use them for the flow your currently working on so you can just add the saved component to your current project. This does mean it needs a maximum of tries to do this specific task as this means that it could create the 'halting problem' inside of the component and retry itself until infinity. This can scrape the returning arrow as everything that needs to be done will be done inside one component.
If we take a look at the second example given you have the functionality to return the information given to a component back to the sender when an error occurs to retry the functionality inside that component. Another way this could work would be by putting it in the error handling that we already have, for now we had the idea when an error occurs it should be sent to an employee. But it could also have an extra space where you can enter how many retries you should have before it actually sends this through. This could completely remove the returning arrow as it isn't needed anymore as it will happen inside the component.

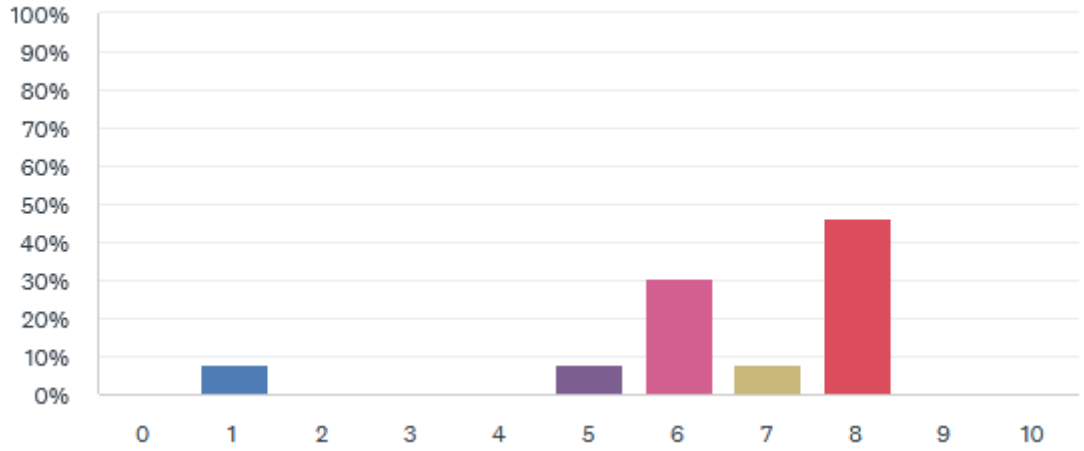
Would a returning arrow be good for the user-friendliness

It is also important to find out if a returning arrow would be user friendly, how can I find out if it'll be user friendly? I have done research about UX and the psychology that is used and referred a lot with UX-design, I will be using these to answer that question (Source 2).
If we look at Jakob's law we see that users usually tend to use the knowledge they have made from other websites/applications to use yours. This means that they expect things to work and be the same as others. When looking at other flow tools they usually tend to go from left to right, never back. This is very logical when thinking of the reading direction, we always read from left to right (with some exceptions in specific countries) so it makes sense that the flow would go from left to right. This is the most easy to read and user-friendly for users as they read this way in almost every scenario.
If we take a look at Hick's Law we see that users take a longer time when the number and complexity of choices is higher. This means, the less the better. So it is important that the flow tool should be simple so the users have a fast understanding on how everything works and can work as optimal as possible. I have also chatted with my stakeholders and they said the exact same to me, they find it important that the application is very simple to understand and easy to use, as it should be appliable for every kind of user, even with little to no understanding of coding.
If we look at the Aesthetic usability effect, we see that users often perceive aesthetically pleasing designs as designs that are more usable. This means that users would prefer good looking applications rather than applications that work better, I believe that a returning arrow would interfere with the aesthetics of the page and would make it less readable for users. This means that the user experience would fall behind for this functionality.

User test

To finish this research I wanted to get some feedback from users, So I looked at the cmd methods, specifically the field research. And I set up a survey for people to answer so I could get more insight on peoples thoughts about a returning arrow. I shared survey to people of various skill levels of software development, I sent it to people that have a lot of experience, people that are just started software developing and people who didn´t have any knowledge at all. I chose this range of people as these will be our target audience for the application.
The survey contains a few general questions on the returning arrow, such as if people know what functionalities could be used with these, or if they think it would be visually pleasing. I got a variety of answers, and I will summarize what the majority of the respondents have filled in. First of all I asked the question how much experience they had with software development, this is an important thing to know before the other questions because then I know how different skilled people look at this problem. The results showed that I mostly had pretty decent software developers but also a few people that didn't have a lot of experience,

this was a good variety which lead to some pretty good answers.



Now for the rest of the questions I will write them down and summarize what has been answered;

***Q1**: "Which functionalities do you think will require a returning arrow?"*
Most answered: "When an error occurs."
Other answers:
-"A kind of return statement, if you want to call something from the previous component."
-"When you need the changed state."
-"When a connection needs to go 2 ways."

***Q2**: "Do you think the 'returning arrow' could cause confusion (since we always read from left to right in the 'western world')?"*
Concrete yes/maybe/no: 30%/10%/60%
Broad answers:
-"No, the system will work in visual blocks, so it doesn't need the western script logic to work"
-"I think we are used to working differently than from left to right."
-"I think it isn't confusing, it would get confusing when you get more switchcase -> module -> switchcase because you need more arrows and it would be possible that you won't understand the order anymore. For that I'd use numbering."
-"Yes, but not because of the western reading manner, I think it's kind of ambiguous."

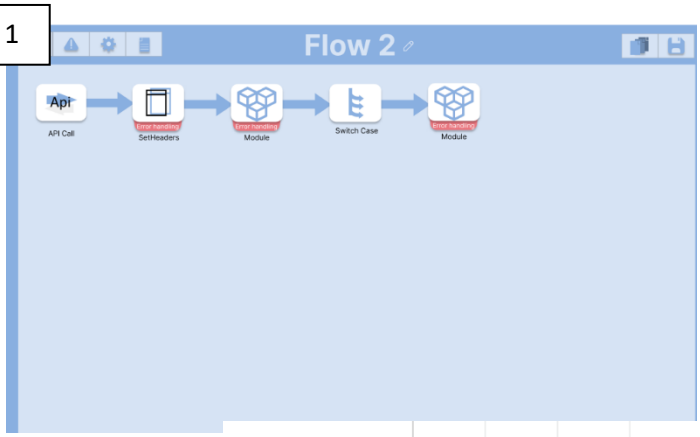
Q3: "*The system should be usable for all types of users, do you think a returning arrow can cause problems for people who have little software experience?"*
Concrete yes/maybe/no: 40%/20%/40%
Broad answers:
-" yes possibly, people with little experience with software often don't immediately think of data lifecycles, and that concept can be quite difficult to understand."

-“I think especially the older generation will not understand. Young people or at least the people with basic technical insight must understand it.”
-“ depends on how it is depicted. I think it's more readable for everyone if the return arrow is separated from the original arrow.”
-“Depending on how it looks, anything can cause problems, but as long as it's clear and organized I don't think so.”

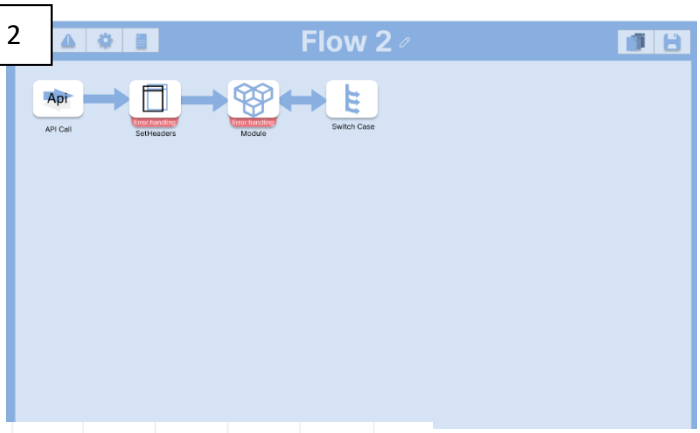
Q4: “*due to a returning arrow it may keep circulating between two blocks, do you think this is a big problem? How could this be resolved?”*
Most answered: “A maximum of returns”
Broad answers:
-“By including a tooltip to prevent this problem, you can warn users in advance.”
-“yes, this will cause everything to crash and you will not be able to give any further inputs. you can solve this by making sure at the backend that there is a limit on the loop or that it is not necessarily a loop but a method that performs an action x times back and forth.”
-“That could cause confusion. You could solve this by showing which action he is currently doing, or by generating a log in which order he performs the actions.”
-“ Yes that's a big problem, that's how you get stack overflow. The user must write code that does not allow this. But the "returning arrow" won't help much.”
-“I think you solve this by not forcing people to use it. Offer them the position, but don't oblige them. When they go back and get further in the app it is strange if the keep going back to change their choice.”

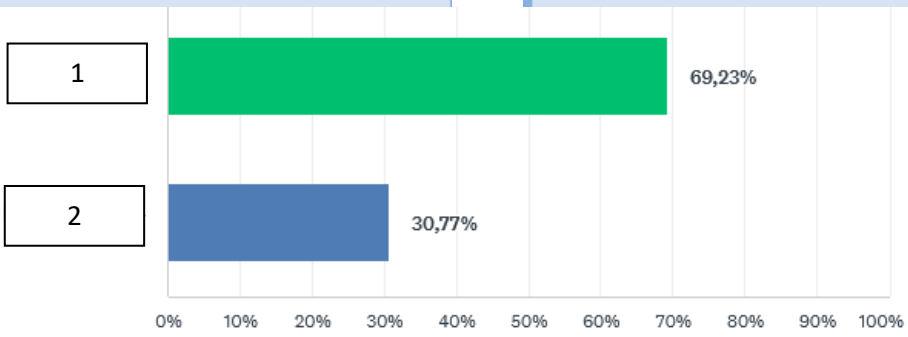
Q5: “If you see these two pictures, which one would have your visual preference?”

1



2





We can conclude this research by saying that people don't really see a problem with a return arrow if it is displayed and explained the right way so that people with less experience get the hang of it and know what kind of errors it offers. Though people don't think the return arrow is a bad idea the majority still voted on the first picture of the visualization, which excludes the return arrow. So that means they prefer it without the returning arrow and just want to have a more lengthier flow, this wouldn´t cause any problems as people can sort the flow their own it won´t be classified as messy.

I also used the research method participant observation, I went out and gave people 2 visualizations of how the flow could go and I asked them for an opinion while they could play with it and ask questions about it, as I thought it was better for the users to experience and see what we're talking about than just having a picture. I didn´t think the survey was enough of a user test I thought it would be a good addition to guerilla test people. You can find the notes I wrote down __

Conclusion

If we look back at the research we did we can conclude that there are definitely moments when a flow could need a returning arrow, but when doing some more research we can conclude that there are several other ways to solve this. In this research we have come to the conclusion that these other ways could work much better for our project and are also much better if we look at the user friendly part of the spectrum. So to answer the main question, should components be able to receive a returning arrow? They could, but it isn't the best practice way to do so. In the future I will be using the solutions I have researched in this document for my project and this helped me a lot with giving a better idea on how I want my flow tool to look.

Sources

1. CMD Methods Pack - find a combination of research methods that suit your needs
2. The laws of UX.docx - Google Docs
3. https://www.servicenow.com/nl/now-platform.html
4. https://www.xmatters.com/features/flow-designer/
5. https://messagetothemoon.nl/telefonie/hosted-3cx/call-flow-designer
6. https://www.mulesoft.com/platform/api/flow-designer-integration-tool
7. https://powerautomate.microsoft.com/nl-be/
8. Introduction to Recursion - Data Structure and Algorithm Tutorials - GeeksforGeeks
9. Halting Problem | Brilliant Math & Science Wiki