Has supporting macros in noir similar to rust ones been discussed? It would be very helpful to avoid aztec3 contract devs the hassle of needing to to do this in every secret function:

fn transfer(//*****/ // **Should eventually be hidden: inputs: pub Inputs,** //*******/ amount: pub Field, sender: pub Point, recipient: pub Point,) -> pub [Field; dep::aztec3::abi::PUBLIC_INPUTS_LENGTH] { context.args = context.args.push(amount); context.args = context.args.push_array(sender.serialize()); context.args = context.args.push_array(recipient.serialize());

```
    context.return_values = context.return_values.push(result);

    context.finish(inputs)
}
```

Being able to code a macro in the noir-aztec3 that would do the following transformation at the ast level would be amazing:

[secret_fn()]

```
fn transfer(
    amount: pub Field,
    sender: pub Point,
    recipient: pub Point,
) -> pub Point {

    result
}
```

to

fn transfer(inputs: pub Inputs, amount: pub Field, sender: pub Point, recipient: pub Point,) -> pub [Field; dep::aztec3::abi::PUBLIC_INPUTS_LENGTH] { context.args = context.args.push(amount); context.args = context.args.push_array(sender.serialize()); context.args = context.args.push_array(recipient.serialize());

```
    let result = transfer_internal(amount, sender, recipient);

    context.return_values = context.return_values.push_array(result.serialize());

    context.finish(inputs)
}

fn transfer_internal(
    amount: Field,
    sender: Point,
    recipient: Point,
) -> Point {

    result
}
```

Maybe since we have an aztec3-specific compiler that uses noir_wasm

, we could have a transformation function as callbacks called by noir_wasm

? For example, if it finds a macro in the code and is executing in the wasm environment it could call back JS to have it modify the tree.

It could be done as a preprocessing step of the noir code but in that case we'd have to parse noir in typescript, which isn't ideal.