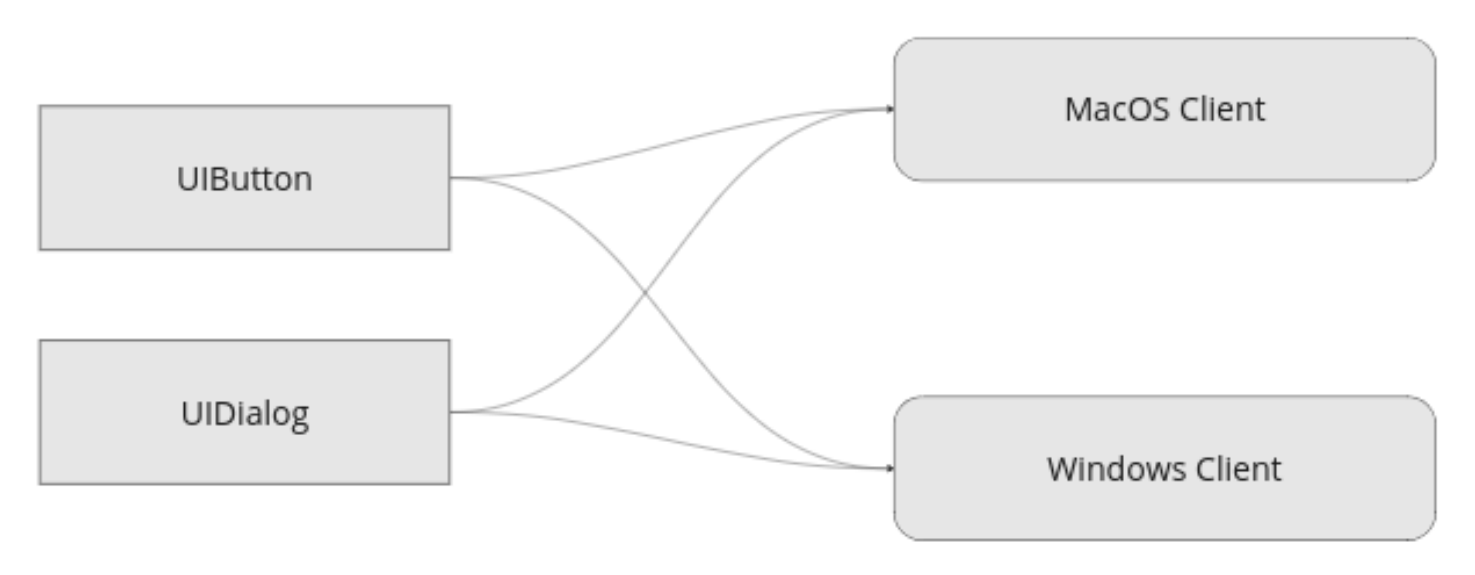
Abstract Factory 💻

Análise e Desenvolvimento de Sistemas - 6º Período

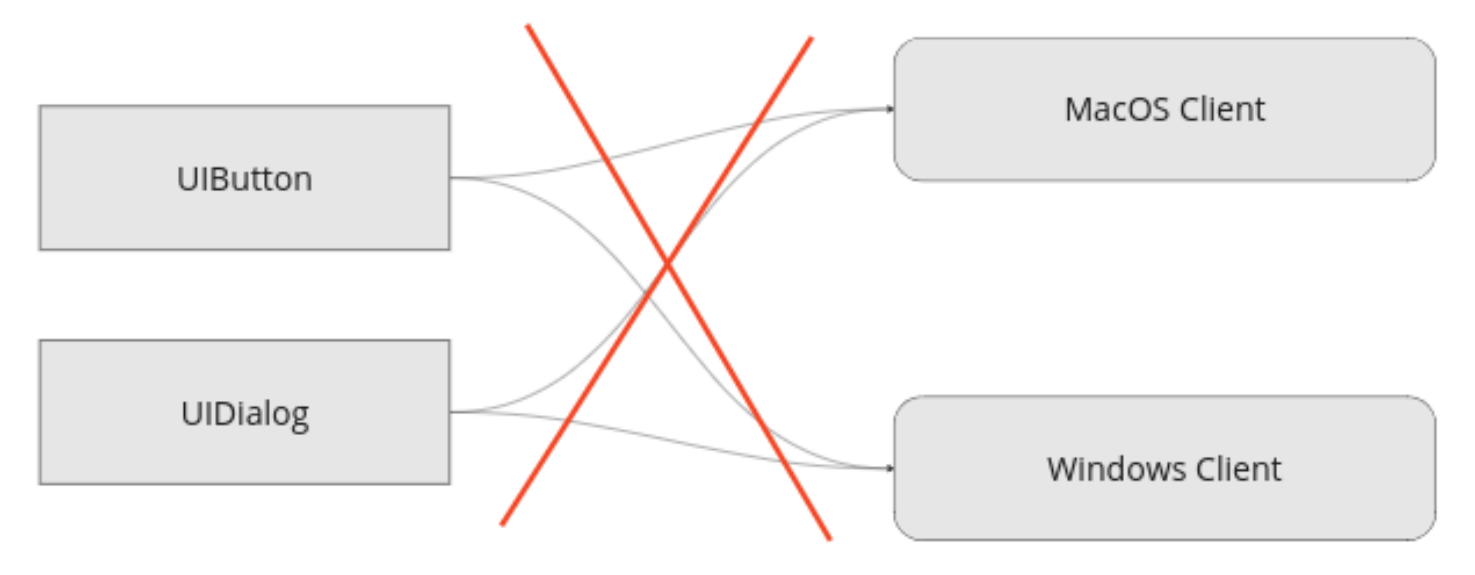
Cenário

- Precisamos de um sistema que crie **elementos de UI (Interface de Usuário) para diferentes SOs**
- Por motivos de compatibilidade, precisamos que os elementos criados sejam da **mesma família (SO)**



Cenário

- A **lógica de criação** dos objetos da mesma família precisa ficar **"escondida"** do client
- Se necessário **alterar o SO, isso deve ser fácil para o cliente**.



Abstract factory

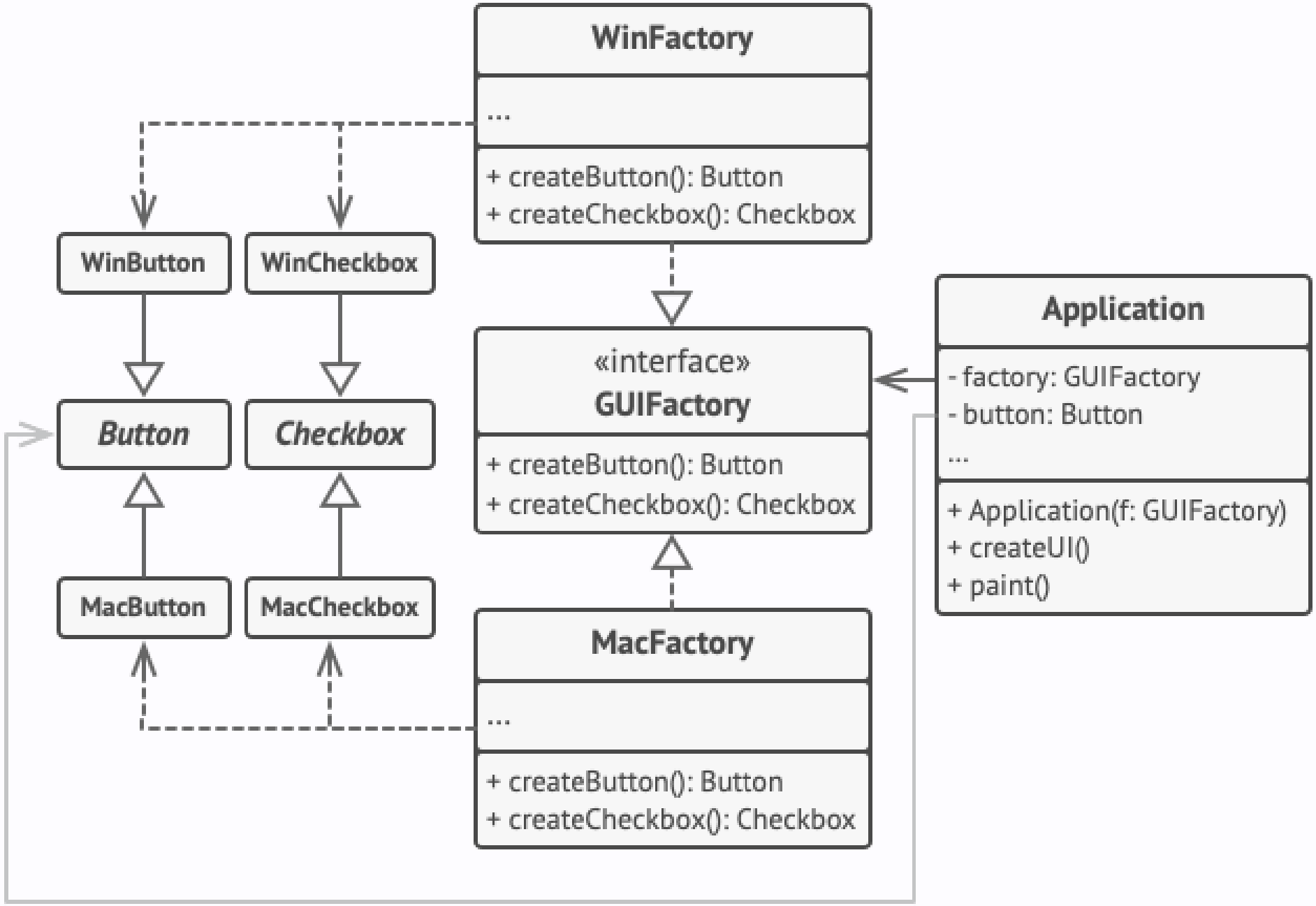
- Providencia uma interface para a criação de famílias de objetos relacionados sem especificar suas classes concretas (**O client não tem acesso as classes concretas**)
- Podemos usar o AbsFactory para criação de fábricas para cada OS (família)
- **Cada fábrica** ficará responsável por criar os elementos corretos para cada OS
- Cria uma **camada de abstração a mais para o factory Method**

Abstract factory

- O **cliente apenas interage com as fábricas**. Se precisar mudar o OS, apenas chama a outra fábrica
- Se alguma manutenção for necessária nos elementos concretos, mudamos nas classes concretas dos elementos (**descoplamento**)

Abstract factory

- Para o **cliente**, qualquer fábrica faz a mesma coisa, e **ele não tem acesso a como esses objetos são definidos** (classes concretas que definem os elementos)
- Ou seja, uma vez que chamamos uma fábrica para determinada família de objetos, **não precisamos nos preocupar com a compatibilidade deles**. Garantimos que serão todos da mesma família


WinFactory
...
+ createButton(): Button
+ createCheckbox(): Checkbox
WinButton
WinCheckbox
Button
Checkbox
«interface»
GUIFactory
+ createButton(): Button
+ createCheckbox(): Checkbox
Application
- factory: GUIFactory
- button: Button
...
+ Application(f: GUIFactory)
+ createUI()
+ paint()
MacButton
MacCheckbox
MacFactory
...
+ createButton(): Button
+ createCheckbox(): Checkbox

Elementos do Abstract Factory

- Interfaces (POO) dos elementos das interfaces
- Implementações concretas dos elementos para cada SO
- Interface que será implementada por cada Fábrica
- Fábricas concretas para cada SO

Elementos do Abstract Factory

- As Interfaces garantem que cada família ficará separada
- Tanto os "produtos", em nosso caso os OSs, quanto as fábricas
- **Importante**: A Interface que as fábricas implementam, na literatura é a **Abstract Factory**. Os objetos que são criados pelas fábricas são chamados de **Produtos**

Resumindo aplicabilidade

- Quando precisamos de famílias de produtos (objetos)
- Separar a criação desses produtos do client (desacoplamento)
- Separar a maneira como cada família contrói seus objetos

Resumindo aplicabilidade

- Open-closed principle. Aberto e fácil de estender para adicionar mais fábricas, sem necessidade de mudar o código que já existe
- Single Responsibility Principle. Separação do código que cria e do que usa. Característica dos padrões de criação.