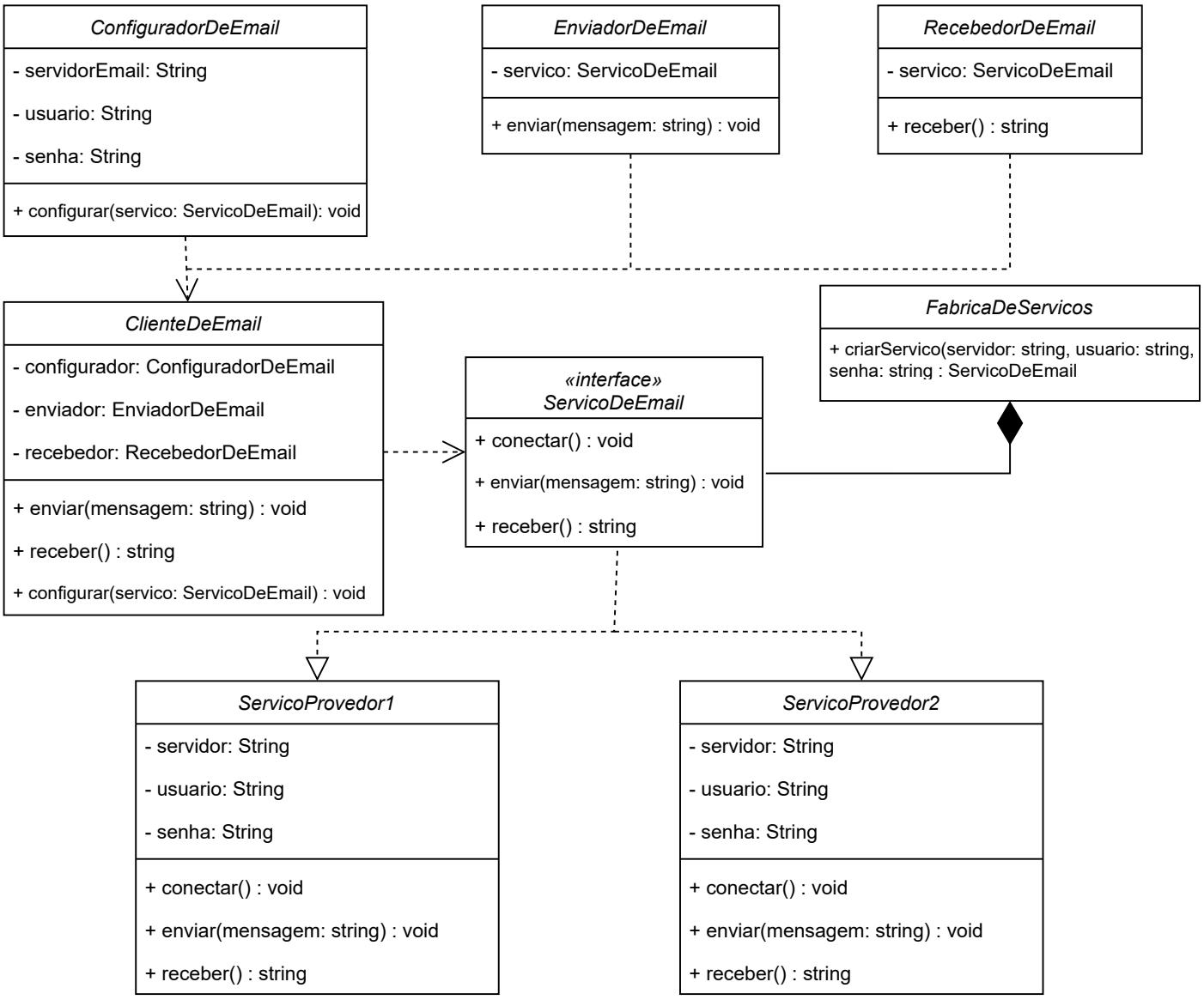

Para atender ao princípio da **Responsabilidade Única (SRP)**, vamos separar as responsabilidades da classe ClienteDeEmail em classes distintas. Teremos as classes **ConfiguradorDeEmail, EnviadorDeEmail e RecebedorDeEmail**. Dessa forma, cada classe terá uma única responsabilidade, o que facilitará a manutenção e a extensibilidade do sistema.

Para respeitar o princípio das classes **Abertas-Fechadas (OCP)**, vamos utilizar o padrão de projeto **Factory Method,** que permite a criação de objetos sem especificar a classe exata a ser criada. Com isso, será possível adicionar novos provedores de e-mail sem precisar modificar o código existente.

Para atender ao princípio da I**nversão de Dependência (DIP),** vamos criar uma interface chamada **ServicoDeEmail**, que será implementada pelas classes ServicoProvedor1 e ServicoProvedor2. A classe **ClienteDeEmail** terá uma dependência apenas da interface **ServicoDeEmail, e não das classes concretas.**

Na solução proposta, temos a classe **ClienteDeEmail** que será responsável por receber as entradas do usuário e utilizar as outras classes para configurar, enviar e receber e-mails.

A classe **ConfiguradorDeEmail** será responsável por configurar a conta de e-mail, podendo receber serviços de qualquer provedor de e-mail que implemente a interface **ServicoDeEmail**.

A classe **EnviadorDeEmail** será responsável por enviar e-mails através do serviço de e-mail configurado, também podendo receber serviços de qualquer provedor de e-mail que implemente a interface **ServicoDeEmail.**

A classe **RecebedorDeEmail** será responsável por receber e-mails através do serviço de e-mail configurado, podendo receber serviços de qualquer provedor de e-mail que implemente a interface **ServicoDeEmail.**

O padrão de projeto **Factory Method** será utilizado nas classes **ConfiguradorDeEmail, EnviadorDeEmail e RecebedorDeEmail**, permitindo a criação de objetos de serviços de e-mail sem especificar a classe concreta.

A interface **ServicoDeEmail** será implementada pelas classes **ServicoProvedor1 e ServicoProvedor2**, que são classes concretas que se relacionam com a classe abstrata (interface).

Com essa solução, podemos respeitar os princípios SOLID e atender aos requisitos do sistema de gerenciamento de e-mails proposto.