God Object
Un God Object en desarrollo de software es cuando una sola clase que conoce muchas cosas o hace muchas cosas (o ambos al mismo tiempo) y por tanto es bastante difícil de testear o de cambiar. Por ejemplo una clase que tenga muchísimos métodos o muchísimos atributos. El tener un God Object hace que sea extremadamente difícil el hacer pruebas por bugs que se entrelazan a la misma clase cuando se debe cambiar algo o se debe aumentar algo a la misma clase. La solución a esto es realizar una refactorización del código y separar en distintos grupos dependiendo de la funcionalidad u otro criterio.

Database as IPC
Otro anti-patrón donde una base de datos se utiliza para la comunicación entre procesos mientras que un mecanismo de IPC ligero es mucho más beneficioso y eficiente.

Batch Over MQ
Utilizar una transferencia de mensajes en tiempo real en conjunto en vez de utilizar un protocolo de cola de mensajes para esta integración.

Fat Class
Este antipatrón ocurre cuando la clase es más “gorda” que profunda. Esto quiere decir que existen clases de las cuales hay varias que heredan de ellas. Esas superclases entonces se vuelven difíciles de cambiar ya que hay muchas de las cuales dependen directamente. Esto hace que existan clases que hacen depender a toda una lógica en el código (estas serian las clases gordas).

Magic PushButton
Finalmente, este antipatrón es común en el desarrollo de ambientes gráficos. Ocurre cuando el programador conoce lo que se debe validar o la lógica de negocio del programa y decide utilizar la interfaz gráfica para implementar la lógica antes que utilizar una clase o algún artefacto separado. Esto hace que el código pueda crecer incontrolablemente, sea difícil de testear y sea difícil agregar alguna cosa al código. Para ello, se debe decidir utilizar el modelo MVC para tener la lógica separada de la interfaz del usuario.