En nuestros proyectos hay clases de estructura muy similar entre ellas, como los repositorios base y los convertidores. Para optimizar la creación de estas clases se han creado unos scripts de python. Estos scripts toman una lista de nombres de clases del modelo de dominio y a partir de un archivo plantilla (en nuestro caso, los repositorios y convertidores de actor), se crean todos los archivos necesarios tomando el archivo plantilla y sustituyendo cualquier instancia del nombre de clase usado en la plantilla por el nombre de la clase del modelo de dominio de la iteración actual (respetando si la primera letra es minúscula o mayúscula). Tras iterar sobre todas las clases del modelo de dominio, se obtienen todas las clases de ese tipo necesarias, idénticas a la plantilla y preparadas para hacer modificaciones adicionales si fuere necesario (por ejemplo, añadir queries en el caso de las clases de repositorio).

Esto ahorra tiempo gracias a ahorrar la necesidad de crear todas estas clases a mano.

El código de estos scripts se adjunta a continuación. Los scripts de ejemplo usa las clases del proyecto Acme-Madrugá y para funcionar requieren los archivos ActorRepository, ActorToStringConverter y StringToActorConverter.

*create_converters.py*

```
lista = ["AcmeFloat", "Administrator", "Area", "Brotherhood", "Finder",
"IsRegistered", "Member", "Message", "MessageBox", "Procession", "Request",
"SocialProfile", "SystemConfiguration", "UserAccount"]

for name in lista:
    template_to = open("ActorToStringConverter.java", "r").read()
    template_to = template_to.replace("Actor", name)
    template_to = template_to.replace("actor", name[0:1].lower() + name[1:])
    open(name + "ToStringConverter.java", "w").write(template_to)
    template_from = open("StringToActorConverter.java", "r").read()
    template_from = template_from.replace("Actor", name)
    template_from = template_from.replace("actor", name[0:1].lower() + name[1:])
    open("StringTo" + name + "Converter.java", "w").write(template_from)
```

*create_repositories.py*

```
lista = ["AcmeFloat", "Administrator", "Area", "Brotherhood", "Finder",
"IsRegistered", "Member", "Message", "MessageBox", "Procession", "Request",
"SocialProfile", "SystemConfiguration", "UserAccount"]

for name in lista:
    template_to = open("ActorRepository.java", "r").read()
    template_to = template_to.replace("Actor", name)
    template_to = template_to.replace("actor", name[0:1].lower() + name[1:])
    open(name + "Repository.java", "w").write(template_to)
```