Jose Enrico B. Tiongson
Title: DragonSMS 2 - Cleaning the Mess
Document: Design Pattern Documentation

1. **Strategy**	
Type	Behavioral
Intent	Enable algorithm behavior to be selected during runtime
Implementing classes `dragonsms.IOStream` • Simple class that contains a selectable `InputStream` and `OutputStream` • Used by `DragonServer` for indefinitely passing messages between two streams • You can change either using `setIn()` / `setOut()`	
2. **Memento**	
Type	Behavioral
Intent	Provides ability to restore state of objects
Implementing classes `dragonsms.session.SessionManager` • Class that manages opening, closing, and saving of user sessions • Allows saving of user's room and game state using `startNewSession()` and `restoreSession()` methods • Connects with hibernate database through the SessionDao and SessionRepository classes	
3. **Prototype + Factory**	
Type	Creational
Intent	Handles creation of new objects via cloning a prototype
Implementing classes `com.elegantsms.util.TypeConverterFactory` • Creates common TypeConverter classes by cloning previously created prototypes • E.g. `TypeConverter<Integer>` is created from `TypeConverterFactory.createConverter(Integer.class)` • The default `TypeConverterMap` is also created by cloning a copy of the collection • Used so that I won't need to create the anonymous objects in the factory method itself, just recycle the previous classes	
• **Builder**	
Type	Creational
Intent	Polymorphically builds an object step-by-step to avoid telescoping constructor anti-pattern

Implementing classes

`com.elegantsms.framework.DispatchMethod` -> `StringBuilder`

- Uses `StringBuilder` to convert the `SmsQuery` pattern to a usable Regular Expression
- So that it's more efficient to create the `String` instead of concatenating everything

5. **Singleton**	
Type	Creational
Intent	Restricts instantiation of a class to one object

Implementing classes

`dragonsms.session.SessionDao`

- Package-private singleton class that is accessible by all `SessionManager`
- Provides a single reference to the `Session` database table through hibernate
- Only one reference is needed even if there are multiple `SessionManagers` accessing the same hibernate instance

6. **Decorator**	
Type	Structural
Intent	Allows behavior to be added to an individual object dynamically

Implementing classes

`com.elegantsms.framework.SmsApplication`

- Allows dynamic addition of new `SmsModules` and `TypeConverters` without altering its structure
- One can simply call `addModule()` and `addTypeConverter()` methods for the application, for instance when one wants to use custom classes

`com.elegantsms.framework.SmsModule`

- Fields can be decorated with `@SmsInjection` so that one can add objects to a module through the SmsApplication owning the module

`com.elegantsms.framework.SmsQuery`

- Can be decorated with `@CaseSensitive`, `@DispatchPriority`, `@RegexDebug`, and `@ArrayDelim`