|  |  |  |
| --- | --- | --- |
| Design Patterns | March 6  2016 | |
| Monica Stoica  Rosen Danev  Toy Factory  Application | | Factory pattern |

Table of Contents

[Introduction 2](#_Toc444544259)

[UML Diagram 2](#_Toc444544260)

[User interface 3](#_Toc444544261)

[Reusability 4](#_Toc444544262)

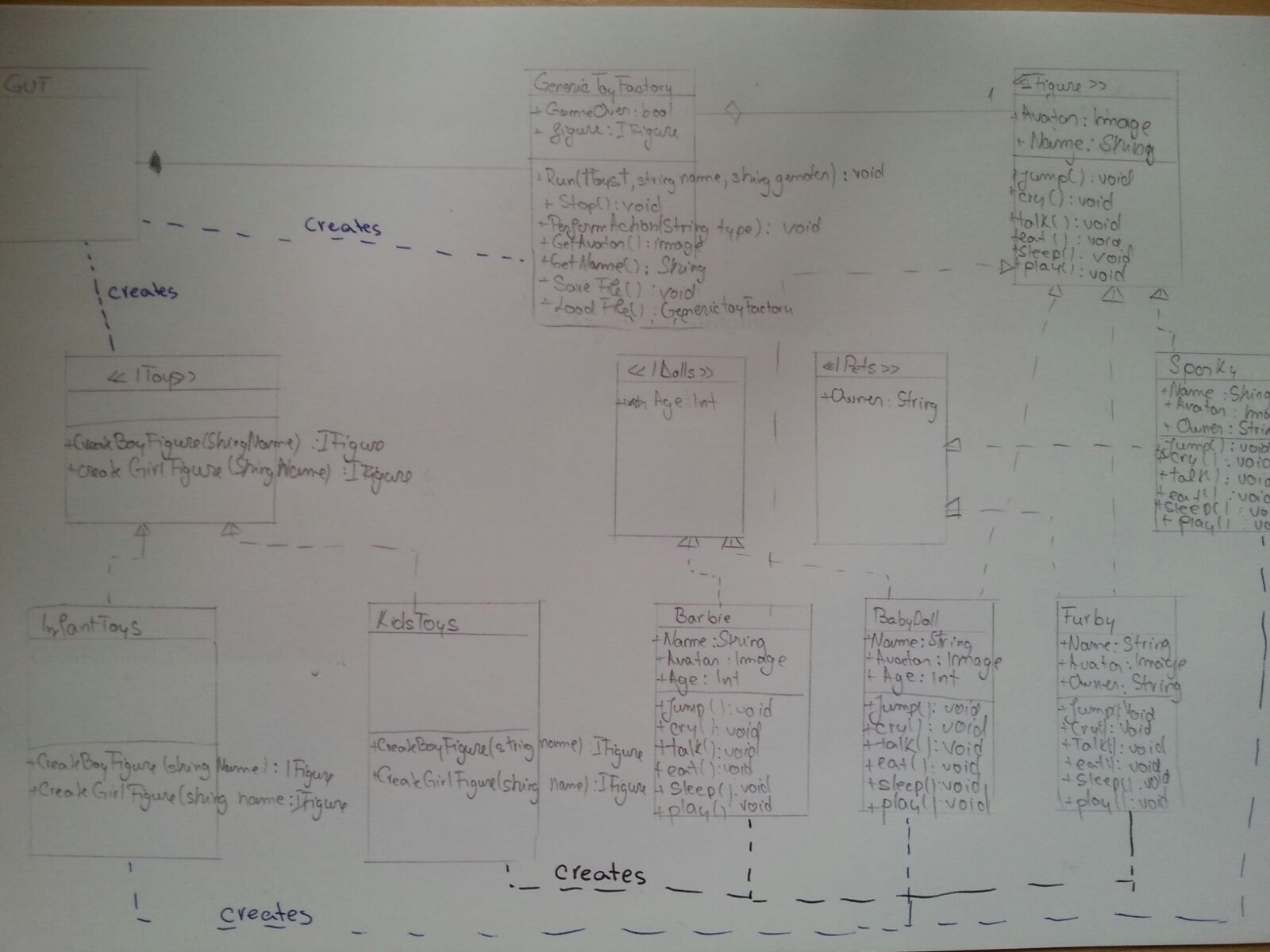
[Maintainability 4](#_Toc444544263)

[Extensibility 4](#_Toc444544264)

# Introduction

The following document analyses the three main characteristics of the pattern used to develop an application that monitories the public transportation such as: trains, buses and trams. This application allows the user to see the schedule of the above named public transports. The user can also add or remove a vehicle and/or delay it. The observer, the traffic monitor is subscribed to both three types of commuting. Therefore, whenever the state of one of vehicles changes, the observer will be notified. The observer can also choose to subscribe/unsubscribe only to the subjects that interest it (e.g Monitor only the trains and unsubscribe for the busses and trams). A pattern is general reusable solution to a commonly occurring problem within a giving context. The purpose of using patterns is to speed up the developing process and helps preventing issues that can cause major problems.

# UML Diagram

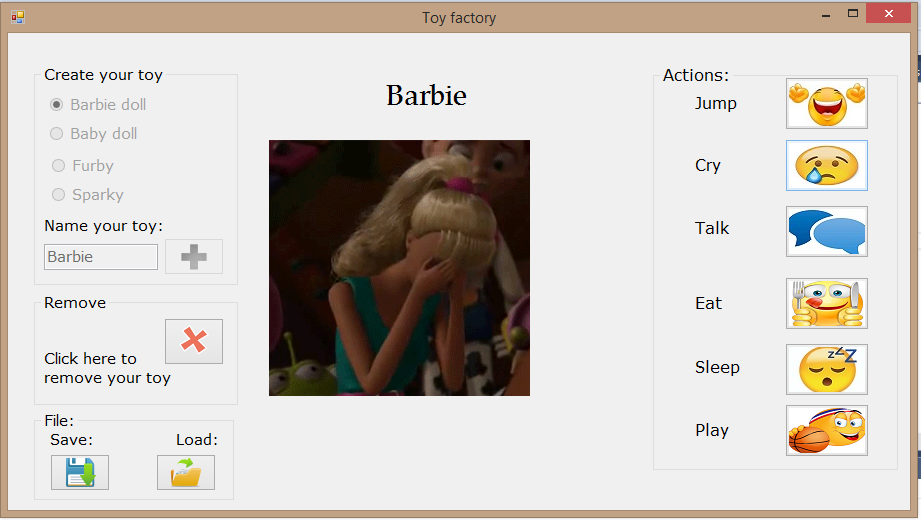


# User interface

The following picture represents the User Interface of the Toy Factory application. The user can create one of the four available toys. He/she enters a name for the toy an presses the button create. Once a toy has been created, another one cannot be added. To create another toy, the current one has to be removed first.

A toy can perform several actions such as jump and talk. Every time the button is pushed, an animation will be shown on the screen and for Cry and Talk actions, sound will be played.

The user can save the toy that he has created and he can load it later to continue playing with it.



# 

# Reusability

One of the main reasons why patterns are useful is because they can be easily reused without changing the code. Considering our above described application, we can attest that the pattern is reusable. The factory pattern can be reused for different types of factories or products without modifying the GenericToyFactory class.

# Maintainability

Because the classes are not tightly coupled and the pattern is quite small, the system is easily maintained. When a system is easy to maintain it means that new features can be added.

# Extensibility

New functionality can be provided by adding new code without changing the initial one. Therefore, there is no need to worry about bugs or causing problems. The current pattern can be easily extended because it does not depend on other classes, the concrete classes being isolated. There can be added as many factories or products as wanted. Moreover, the IFigure interface can be extended by adding different actions for the toys. However, because the products, in this case Barbie and BabyDoll have similar behaviors, it is difficult to add a new product which performs different actions but is still a doll.