Module « T-CPP-710 »

-

Project « Plazza »

Software Architecture Specifications (SAS)

Table of contents

OBJ

Revision history

Date	Version	Description	Author
17-12-2017	1.0	Software Architecture Document generated	Julie NGUYEN

Introduction

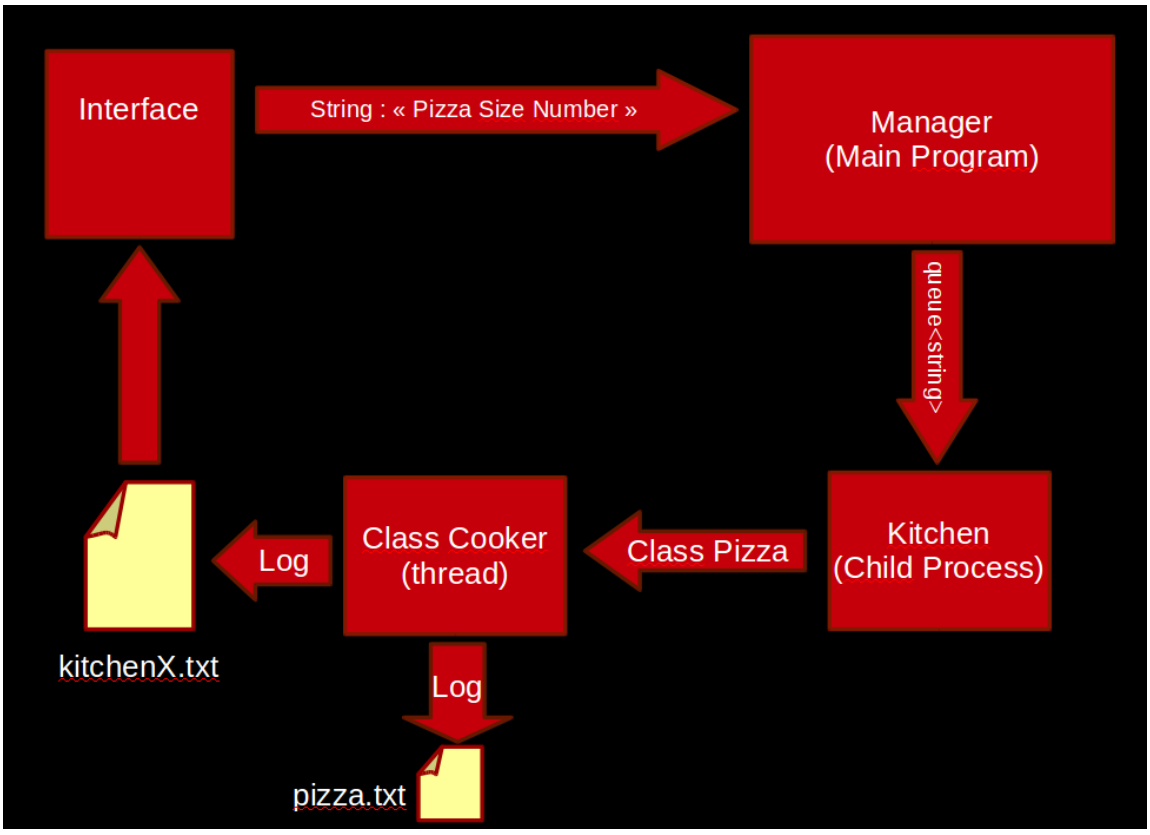
The aim of this software architecture specification (SAS) is to present the technical elements necessary for the Plazza project. This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1. Project context

The project consists of a simple system to take commands for EpiPizza, a multinational company which sends Pizza. The program must consider that the company has different numbers of kitchens and cookers depending on its branches, and therefore equilibrate the orders.

2. Global architecture

Global view of the project, what are the different components and how they interact between each other.



Components interactions

3. Components description

a. Interface

DESCRIPTION

The technology used for the interface is **NCurses** in order to answer to the IHM requirement *REQ_IHM_010*: a simple CLI system is used with the line command.

There is a window divided into several parts, which are updated separately or together according to the user's choices.

The user chooses which pizzas he wants to : their type (Regina, American, Fantasia or Margarita), their size (M, L or XL) and how many of them he wants to.

The interface saves those choices, which will then be transmitted to the back-end of the application with a simple std::string.

The user can choose to see the status of the kitchens and the cookers at the end of his orders (IHM requirement *REQ_IHM_020*).

CLASSES

- WindowFront.h
- WindowFront.cpp

b. Manager

Description of component 2: utility, main classes, key functions and actions, link to other components.

DESCRIPTION

The manager is the heart of this application. This class translates the orders made by the user through the interface (simple std::string into a std::queue<String>), and then creates the necessary processes (in other word, the kitchens) to complete theses orders, and affixes the necessary threads (here the cookers) to the processes.

The manager can evaluate the number of busy or free cookers in the kitchens too.

CLASSES

- Manager.h
- Manager.cpp

4. Traceability matrix

This matrix make the correspondence between components, classes, functions, and requirements developed in the request for proposal:

Id requirement	Requirement description	Component	Function / action
REQ_DESIGN_010	Your system must implement thread using std::thread	Cooker	Void runThread([parameters]); Void cookPizza([parameters]);
REQ_DESIGN_020	You must propose thread pool system	Team	
REQ_DESIGN_030	You must propose queue system for orders	Kitchen	Std::queue<std::string>orders attribute
REQ_DESIGN_040	You must separate home process command from kitchen. You must have one process for home and command and one for each kitchen	Manager and Kitchen classes	void Manager::manageKitchens([paramaters]);
REQ_DESIGN_050	The cookers are symbolized by thread	Cooker	
REQ_DESIGN_060	The kitchens process must be created on demand	Manager and Kitchen classes	void Manager::manageKitchens([paramaters]); Count how many kitchens are needed depending of the orders
REQ_FUNC_010	User should write new orders on standard entry	WindowFront	Void createCurses(); WINDOW *createUserwin([paramaters]);

REQ_FUNC_020	Orders are dispatched betweend kitchens and cookers fairly	Manager and Kitchen classes	void Manager::manageKitchens([paramaters]); Count how many kitchens are needed depending of the orders
REQ_FUNC_030	You must consider this list of pizzas : Margarita, Regina, American, Fantasia	PizzaFactory	Enum PizzaType
REQ_FUNC_040	You must consider this list of sizes : M, L, XL	PizzaFactory	Std::unique_ptr<APizza> createPizza(PizzaType, std::string const &size); Parameter size is considered
REQ_FUNC_050	You must consider this syntax for orders •S := TYPE SIZE NUMBER [;TYPE SIZE NUMBERS]* •TYPE := Margarita |Regina |American |Fantasia • SIZE := M|L|XL • NUMBERS := x[1..9][0..9]*	Order and Manager	Std::queue<std::string> Manager::convertInputIntoOrder([parameter]);
REQ_FUNC_060	Each Pizza have time to cook: •Margarita: 1.5T •Regina: 1.0T	APizza American Fantasia Margarita Regina	Float getCookTime();

	•American: 2.0T •Fantasia: 4.0T		
REQ_FUNC_070	The base time (T) is given on launch of program (time in millisecond format)	main	Int main([parameters]);
REQ_FUNC_080	The numbers of cookers by kitchen is given on launch of program	main	Int main([parameters]);

REQ_FUNC_090	Kitchens are created on demand and destroyed when no command is send within 5T on this kitchen	Kitchen
		Manager
			void Manager::manageKitchens([param]);
			void Kitchen::updateStatus(int timeBase);
			void Kitchen::quit();() {

REQ_IHM_010	Implement simple CLI system with line command	WindowFront	Void createCurses(); WINDOW *createUserWin([param]);
REQ_IHM_020	You must have kitchens and cookers occupation view	WindowFront	WINDOW *createKitchenWin([param]);