MaroonPrint
Software Architectural Design

Submitted to:

Asst. Prof. Ma. Rowena C. Solamo
Faculty Member
Department of Computer Science
College of Engineering
University of the Philippines, Diliman

Submitted by:
Lee, Kristine-Clair
Magno, Hannah Mae
Wu, Jeremy Jin Qian

In partial fulfillment of Academic Requirements
for the course
CS 191 Software Engineering I
of the
1st Semester, AY 2018-2019

Unique Reference:

The documents are stored in the https://maroonprint.tumblr.com/project-deliverables referenced with MaroonPrint - Architectural Design.

Purpose:

The purpose of this document is to define the consolidated classes from the previous documents to help the developers finalize the software architecture of the system.

Audience:

University of the Philippines Diliman engineering students, faculty, and other personnel and also people who are assigned in maintaining the fire exits.

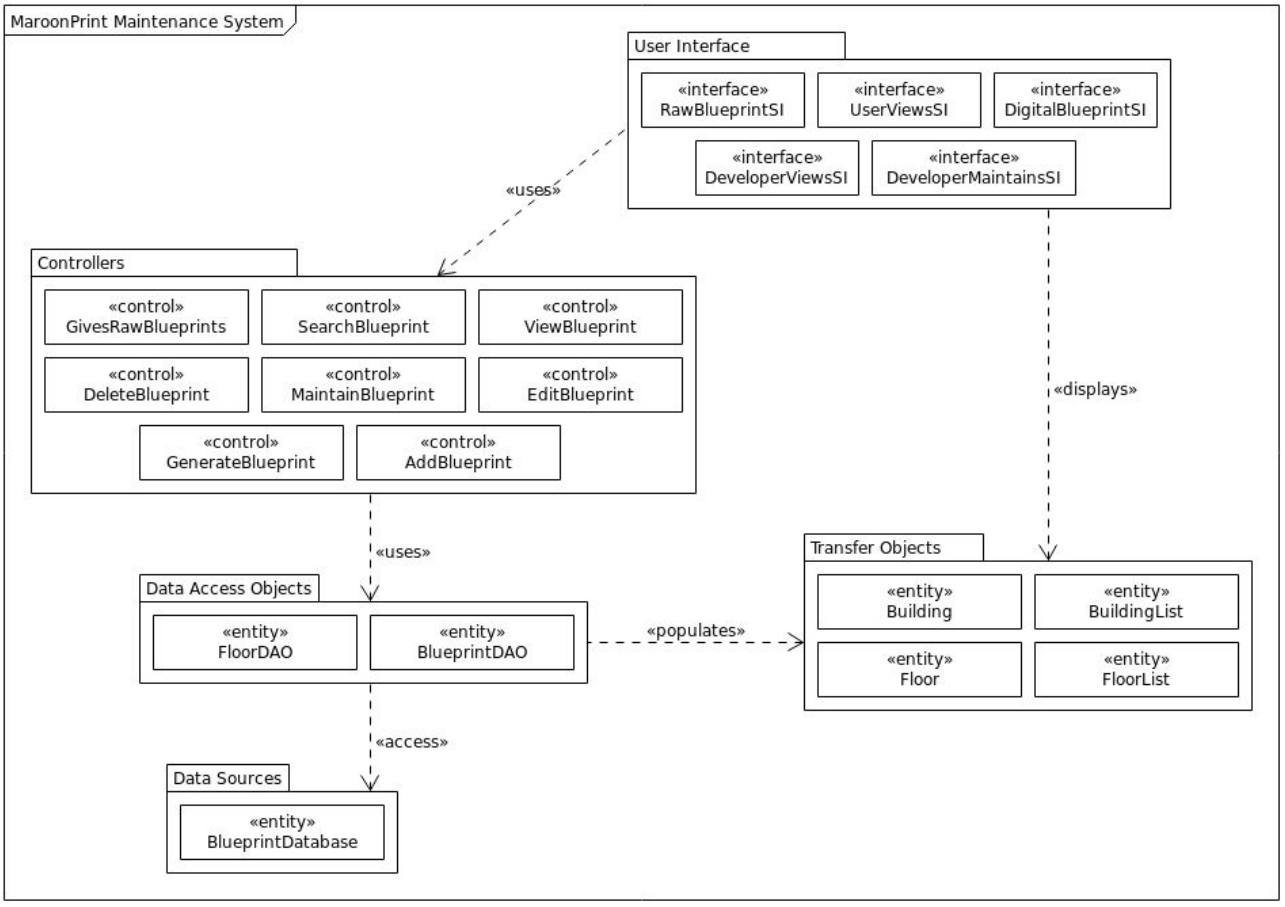
Revision Control:

Revision Date	*Person Responsible*	*Version Number*	*Contribution/Modification*
11/02/2018	Kristine-Clair Lee	1.0	Initial Document; Added diagram; Added contents for transfer object packages
11/02/2018	Hannah Mae Magno	1.1	Added the contents for controller and data sources package
11/02/2018	Wu, Jeremy Jin Qian	1.2	Added contents for User Interface package. Edited format.
02/06/2019	Hannah Mae Magno	1.3	Changed the contents for Data Access Objects Packages (DAO)
02/20/2019	Kristine-Clair Lee	1.4	Updated UML diagram and entities

***System Name*:** MaroonPrint Maintenance System

Description: There are 5 boundary classes in the analysis model, namely, DeveloperMaintainSI, DeveloperViewUI, UserViewsUI, RawBlueprintUI, and DigitalBlueprintSI. In addition, there are 8 control classes which are GenerateBlueprint, SearchBlueprint, ViewBlueprint, GiveRawBlueprint, MaintainBlueprint which is an abstract class extended to DeleteBlueprint, AddBlueprint, and EditBlueprint. Lastly, there are 2 entity classes, Blueprint and RawBlueprint.

***Revised Software Architecture Model*:**



User Interface Package:

Class Name	**Description**
RawBlueprintSI	This is the system interface of the admin to the system whenever he or she needs to provide blueprint. *Attributes:* private Int rawBlueprintID; private String rawBlueprintName; private Rawblueprint r; *Responsibilities*: public void enterRawBlueprint(int rawBlueprintID, String rawBlueprintName) public void givesRawBlueprint(RawBlueprint r)
UserViewsUI	This is the user interface of the user to the system whenever he or she needs to view blueprint. *Attributes:* private Blueprint b; private String bluepritntName; *Responsibilities*: public void clickBlueprintToView(Blueprint b) *//user clicks blueprint to view* public void enterBlueprintToSearch(String blueprintName)*//user searches blueprint to view* public Blueprint getBlueprint(Blueprint b) *//system gets the information given by the user* public Blueprint getBlueprint(String blueprintName) *//system gets the information given by the user*
DeveloperViewsUI	This is the user interface of the developer whenever he or she needs to view blueprint. *Attributes:* private Blueprint b; private String bluepritntName; *Responsibilities*: public void clickBlueprintToView(Blueprint b) *//user clicks blueprint to view* public void enterBlueprintToSearch(String blueprintName)*//user searches blueprint to view* public Blueprint getBlueprint(Blueprint b) *//system gets the information given by the developer* public Blueprint getBlueprint(String blueprintName) *//system gets the information given by the developer*

DeveloperMaintainsSI	This is the system interface of the developer whenever he or she needs to maintain blueprint. *Attributes:* *Responsibilities*: private Blueprint b; private Int blueprintID; private String blueprintName; private String blueprintUpdatedata; private String blueprintdescription; public void enterMaintainBlueprintData(int blueprintID, String blueprintName, String blueprintUpdate, String blueprintDescription) public void submitBlueprintData(Blueprint b)
DigitalBlueprintSI	This is the system interface of the developer whenever he or she needs to generate digital blueprint *Attributes:* private RawBlueprint r; private Blueprint b; private String blueprintUpdatedata; private String blueprintdescription; *Responsibilities*: public void enterDigitalBlueprintData(RawBlueprint r, String blueprintUpdate, String blueprintDescription) public void generatesDigitalBlueprint(Blueprint b)

Controllers Package:

Controller Name	**Description**
GiveRawBlueprint	This is the control that gives the raw blueprint to be generated. *Attributes:* private RawBlueprint r; *Responsibilities*: public void GiveRawBlueprint(RawBlueprint r)
ViewBlueprint	This is the control that views the blueprints in the system. It extends SearchBlueprint *Attributes:* private Blueprint b; *Responsibilities*: public Blueprint ViewBlueprint(Blueprint b)
SearchBlueprint (extends ViewBlueprint)	This is the control that searches the blueprints in the system. It is extended by ViewBlueprint. *Attributes:* private Blueprint b; private String blueprintName; *Responsibilities:* public Blueprint SearchBlueprint(String blueprintName)
MaintainBlueprint (abstract)	This is the control that maintains the blueprints in the system. It is considered an abstract class. *Attributes:* private Blueprint b; private String blueprintName; private String blueprintUpdatedata; private String blueprintdescription;
AddBlueprint (extends MaintainBlueprint)	This is the control that adds a blueprint to the system. It extends MaintainBlueprintController. *Attributes:* private Blueprint b; *Responsibilities*: public void AddBlueprint(Blueprint b)
DeleteBlueprint (extends MaintainBlueprint)	This is the control that deletes a blueprint in the system. It extends MaintainBlueprintController. *Attributes:* private Blueprint b; *Responsibilities*: public void DeleteBlueprint(Blueprint b)

EditBlueprint (extends MaintainBlueprint)	This is the control that edits a blueprint in the system. It extends MaintainBlueprintController. *Attributes*: private Blueprint b; private String blueprintName; private String blueprintUpdatedata; private String blueprintdescription; *Responsibilities*: public void EditBlueprint(Blueprint b,String blueprintName, String blueprintUpdatedata, String blueprintDescription)
GenerateBlueprint	This is the control that generates the digital version of the given raw blueprint. *Attributes*: private Blueprint b; *Responsibilities*: public Blueprint GenerateBlueprint(Blueprint b)

Data Access Objects Packages:

DAO Name	**Description**

BuildingDAO	This is the entity building which contains the data about the blueprint of the building.
	Attributes:
	private char BuildID;
	private String BuildName;
	Methods:
	private void addBlueprint(Blueprint b);
	private void updateBlueprint(Blueprint b);
	private void deleteBlueprint(Blueprint b);

FloorDAO	This is the entity floor which contains the data about the floor of the building. *Attributes*: private char BuildID; private char FloorID; private int FloorNo; private url floorImageLink;

Transfer Objects Package:

Class Name	Description

Building	This is the entity class Building, which contains the data about the Building..
	
	Attributes:
	private char buildID;
	private String buildName;
	
	Methods:
	private void setBuildID(char bid);
	private void setBuildName(String bName);
	...
	private int getBuildingID();
	private String getBuildingName();

BuildingList	This is the entity class BuildingList, which contains the data about the BuildingList
	
	Attributes:
	private ArrayList<String> blueprints
	
	Methods:
	private void addBuildingToList(int bid, String bName);
	…
	private int getCount();
	private ArrayList<String> getList();

Floor	This is the entity class Floor, which contains the data about the Floor..
	
	Attributes:
	private char floorID;
	private String floorName;
	private int FloorNo;
	private url floorImageLink;
	
	Methods:
	private void setFloorID(char bid);
	private void setFloorName(String bName);
	private void setFloorNo(int x);
	private set floorImageLink(url x);
	...

	private int getFloorID();
	private String getFloorName();
	private url getFloorImageLink();

BuildingList	This is the entity class BuildingList, which contains the data about the BuildingList
	
	Attributes:
	private ArrayList<String> blueprints
	
	Methods:
	private void addBuildingToList(int bid, String bName);
	…
	private int getCount();
	private ArrayList<String> getList();

Data Sources Package:

File Name or Database Name	Description
BlueprintDatabase	This is the database that contains all the blueprint records for the MaroonPrint system.