

## **EXAMPLE – DOMAIN ARCHITECT**

This document contains the example of SMartyPerspective technique for the Domain Architect (DAC) scenario by detection of diagram defects with embedded defects of the Software Product Line (SPL) Mobile Media component diagram.

### **1. Mobile Media Software Product Line**

Mobile Media (MM) is an SPL composed of applications (products) that manipulate music, videos and photos for mobile devices such as smartphones and tablets. Provides support for managing (creating, deleting, viewing, executing and sending) different types of media (Young, 2005; Geraldi and OliveiraJr, 2017).

Part of the description of the Mobile Media use cases was taken from the work of Choma (2017) and are presented as follow:

**UC1:** Log in

**Category:** mandatory

**Description:** User enters login and password and the system validates the data.

---

**UC2:** Send Media

**Category:** optional

**Description:** User selects and sends a certain media to another device.

---

**UC3:** Manage Album

**Category:** mandatory

**Description:** User performs general control of album entries. It encompasses the operations of creating, deleting and listing albums.

---

**UC4:** Manage Media

**Category:** mandatory

**Description:** User performs general control of media records. It encompasses the operations of creating, deleting and listing records.

---

**UC5: Manage Favourite Media**

**Category:** optional

**Description:** User performs control of favorite media. It encompasses the operations of setting favorite media and listing favorite media.

---

**UC6: Play Media**

**Category:** mandatory

**Description:** User execute a media according to the possibilities of the product.

---

**UC7: Add Media to Album**

**Category:** mandatory

**Description:** User selects a media and links it to a certain album stored in the system.

---

**UC8: Link Media with Address Book Entry**

**Category:** optional

**Description:** User links media to a particular contact so that it plays when there is a call from that contact.

---

**UC9: View/Hear Media from Incoming Caller**

**Category:** optional

**Description:** When receiving a call, the system plays the media linked to the contact.

---

**UC10: Label Files**

**Category:** optional

**Description:** User inserts a label for a certain media, which can be video, photo or music.

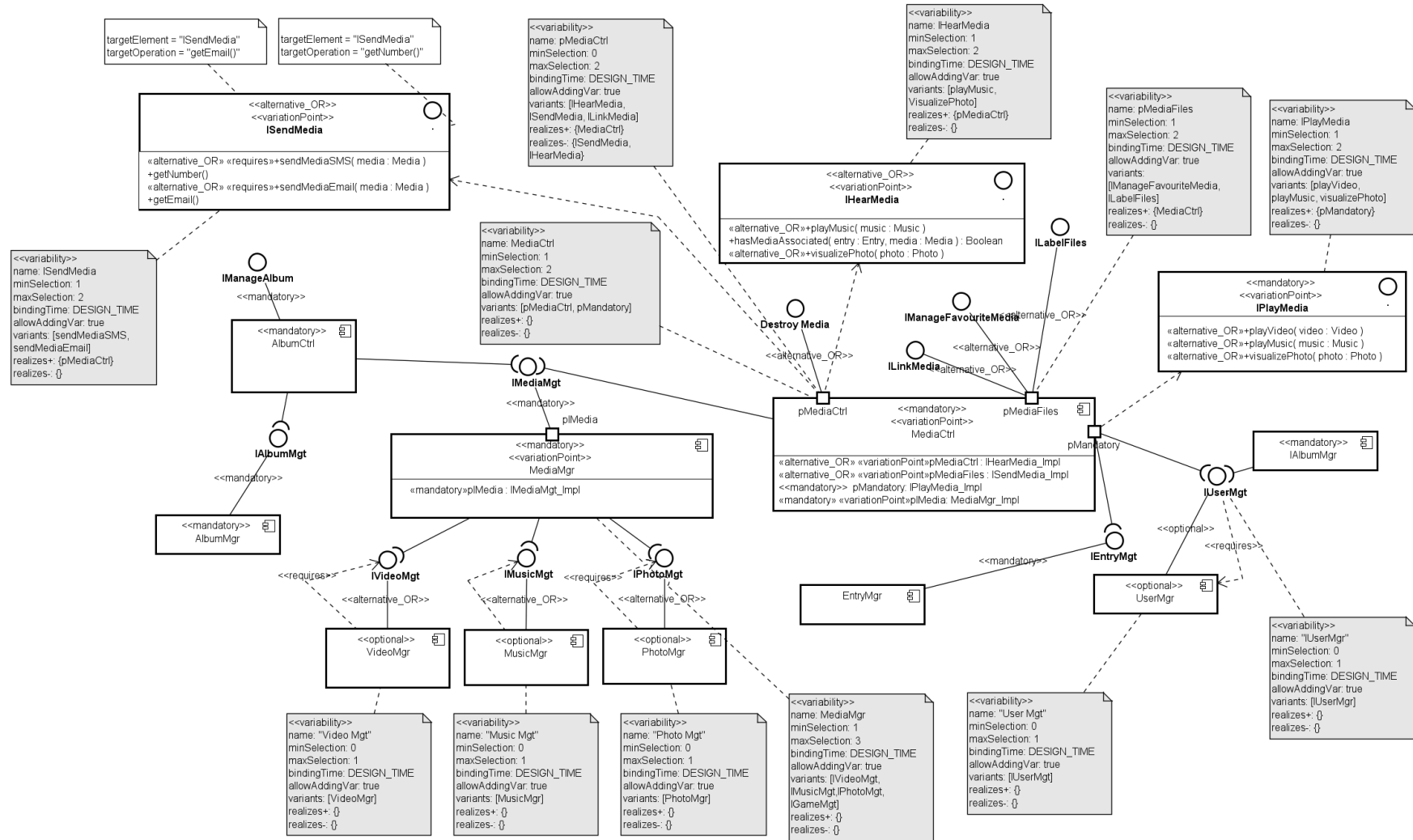
---

## **2. Component Diagram**

Figure 1 show the component diagram for Mobile Media with embedded defects. Use the scenario for your perspective described in Document 5 of this instrumentation to inspect this diagram for defect detection.



**Figure 1: SPL Mobile Media component diagram with embedded defects**



### 3. Resolution for component diagram

The component diagram is an important artifact for the DAC as it expresses how system classes are grouped into components and how they relate to each other through interfaces, thus representing the product line reference architecture.

The DAC scenario starts with the instruction, which introduces the importance of the diagram to the role and then asks the reader to make a list from the class diagram of the system's components and interfaces. Therefore, a correct class diagram (already inspected) must be considered, so wrong information is not considered when inspecting the component diagram.

The reader is free to list the stereotype tags in whatever way is safe for them to understand while reading the diagram. Figure 2 presents an example of a list of classes and interfaces for the Mobile Media example in Figure 1.

**Figure 2:** MM List with components and interfaces

<i>AlbumCtrl</i>	-	mandatory
<i>IManageAlbum</i>	-	mandatory
<i>IaddMediaAlbum</i>	-	mandatory
<i>AlbumMgr</i>	-	mandatory
<i>IAlbumMgt</i>	-	mandatory
<i>IMediaMgt</i>	-	mandatory
<i>MediaMgr</i>	-	mandatory, variationPoint
<i>MusicMgr</i>	-	optional
<i>IMusicMgr</i>	-	alternative_or
<i>PhotoMgr</i>	-	optional
<i>IPhotoMgt</i>	-	alternative_or
<i>VideoMgr</i>	-	optional
<i>IVideoMgt</i>	-	alternative_or
<i>EntryMgr</i>	-	mandatory
<i>IEntryMgt</i>	-	mandatory
<i>MediaCtrl</i>	-	mandatory
<i>ILinkMedia</i>	-	mandatory
<i>IPlayMedia</i>	-	mandatory
<i>IHearMedia</i>	-	mandatory
<i>IManageMedia</i>	-	mandatory
<i>ICabelFiles</i>	-	mandatory
<i>SendMgr</i>	-	optional
<i>ISendMedia</i>	-	optional
<i>FavouriteMgr</i>	-	optional
<i>IManageFavouriteMedia</i>	-	optional
<i>UserMgr</i>	-	mandatory
<i>IUserMgt</i>	-	mandatory

With the list ready, the inspector should follow the steps defined for this diagram and perspective. As previously stated, the reading of the component diagram for the DAC must be

done in 2 steps (Step 3 and 4): the first goes through all the elements (components or interfaces) and the second checks specificities of variability and element management described in classifier format.

For each element of the diagram, the inspector must read all the questions in the step, and only then proceed to the next element, except when the question directs that the reader can stop the inspection and proceed to the next element, or that they are issues that do not fit the element. For example, a subset of questions for optional elements and the element under inspection is mandatory.

The inspection of the component diagram for DAC starts in Step 3. It must be completed for all elements of the diagram (Figure 1), whether it is component or interface type, and only then, proceed with the scenario (Step 4), as , the next step may need information that has already been analyzed previously, such as question 4.3 that detects omission of elements, by checking those that have not yet been crossed out from the list and, consequently, were not described in the diagram.

The reader is free to start the inspection by whichever element they prefers. For this example, the inspection was started from the **IManageAlbum** interface, however, after following the entire procedure in Step 1, that is, reading all the questions in this step, no inconsistencies were found between the component and class diagram, as, for all questions the answer was negative, that is, no defects were found in the element.

**MusicMgr** was the next element inspected for the example. In question 3.6.3 of Step 3, after going through the previous questions, the inspector is asked if the element that requires another was stereotyped with `<<requires>>`. When analyzing the diagram, the **MusicMgr** component will only be on the diagram if the **IMusicMgt** interface is also. Therefore, there is a need to stereotype the relationship with `<<requires>>`, which was not done in the diagram.

When finding a defect, the inspector must fill in the information in the DIF correctly so that later the element is found and the diagram is corrected. The defect found in **MusicMgr** was described in the DIF in the first line (Table 1) with the following information:

- Diagram: component;
- Question Number : 3.6.3
- Element: MusicMgr;
- Identified Defect: The relationship between MusicMgr requires the IMusigMgt interface, and this relationship has not been stereotyped with `<<requires>>`

Continuing the inspection, the reader will identify in **DestroyMedia** the next defect for the example by reading question 3.3. The question analyzes whether the element under inspection is present in the list made by the inspector. However, when analyzing Figure 2, none of the items on the list match this functionality, so a defect was identified and filled in the FID form as follows (Table 1 - line 2):

- Diagram: component;
- Question Number : 3.3
- Element: DestroyMedia;
- Identified Defect: Destroy Media is not a system feature, so there is no interface for it.

After visiting all elements in the component diagram, Step 3 of the technique is completed. Step 4 is composed of two subgroups and one question alone. The inspector must first analyze the variability representations in the UML note related to the elements and in the subgroup 4.2 components and interfaces described in the classifier format.

A specific type of defect for the component diagram is the lack of description of the variants of an element that represents a point of variation and that was described in classifier format, as guided by the CP6 guideline of the SMarty approach. This type of defect is found in the example in Figure 1 in the **MediaMgr** element.

In the **MediaMgr** component, alternative elements for the variation point were not specified in the operations, which in this case include: **IPhotoMgt**, **IVideoMgt** and **IMusicMgt**. Because it is the same type of defect (same question) for the same element, the omission of the 3 interfaces in a single line of the form can be described.

- Diagram: component;
- Question Number : 4.2.2;
- Element: MediaMgr
- Identified Defect: Alternative interfaces: IPhotoMgt, IVideoMgt, IMusicMgt have not been defined in the element compartment.

The last question, 4.3, is analyzed after inspection of subgroups 4.1 and 4.2, which guided the reader to identify the stereotypes of the variants of optional elements and points of variation, as well as their description in the classifier compartment. In this question, the reader should look at the list with the features described and check if any of the items was not represented by any element of the component diagram, thus characterized as an omission defect.

For this example, 2 elements were not defined in the component diagram as shown in Figure 3. Therefore, the omission of these items according to question 4.3 should be noted in the DIF form. For the first item in the list (**IAddMediaAlbum** - Figure 3), this defect was reported in the DIF form (Table 1 - line 11) as follows:

- Diagram: component;
- Question Number : 4.3;
- Element: IAddMediaAlbum;
- Identified defect: An interface for adding media to the album has not been defined.

With the inspection completed and the DIF completed (Table 1), the inspector can move to the next diagram to be inspected.



**Figure 3:** MM List with interfaces and components after inspection of the component diagram for DAC

<i>AlbumGraf</i>		
<i>IManageAlbum</i>		mandatory
<b>→ <i>IaddMediaAlbum</i></b>	-	mandatory
<i>AlbumMgr</i>		mandatory
<i>IAlbumMgr</i>		mandatory
<i>IAlbumMgr</i>		
<i>MediaMgr</i>		mandatory variationPoint
<i>MusicMgr</i>		
<i>IMusicMgr</i>		information_or
<i>PhotoMgr</i>		optional
<i>IPhotoMgr</i>		information_or
<i>PhotoMgr</i>		optional
<i>IPhotoMgr</i>		information_or
<i>Emergency</i>		mandatory
<i>Emergency</i>		mandatory
<i>AlbumGraf</i>		
<i>IAlbumGraf</i>		mandatory
<i>IAlbumGraf</i>		mandatory
<i>IAlbumGraf</i>		
<i>IManageMedia</i>	-	mandatory
<i>AlbumGraf</i>		mandatory
<i>AlbumGraf</i>		
<i>IAlbumGraf</i>		
<i>AlbumGraf</i>		optional
<i>IManageFavouriteMedia</i>	-	optional
<i>AlbumGraf</i>		mandatory
<i>AlbumGraf</i>		



**Table 1:** DIF after inspection of component diagram for DAC

nº	DIAGRAM					QUESTION NUMBER	ELEMENT	IDENTIFIED DEFECT
	FT	UC	CL	CP	SQ			
1				X		3.6.3	IMusicMgt	The relationship between MusicMgr requires the IMusicMgt interface, and this relationship has not been stereotyped with <<requires>>
2				X		3.3	Destroy Media	Destroy Media is not a system feature, so there is no interface for it.
3				X		3.6.1	ILinkMedia	The ILinkMedia interface is not related to pMediaFiles port, but rather to pMediaCtrl.
4				X		3.2	IAlbumMgr	The component is duplicated with another one already defined in the diagram.
5				X		3.4	EntryMgr	The component stereotype has not been defined in the diagram.
6				X		4.1.1	MediaMgr	IGameMgt defined in variants in variability notation is not a variant for the component.
7				X		4.1.4	IPlayMedia	The component has 3 variants, but maxSelection has been set to 2.
8				X		4.2.2	MediaMgr	Alternative interfaces have not been defined in the element compartment: IPhotoMgt, IVideoMgt, IMusicMgt
9				X		4.2.3	MediaCtrl	It has been defined in the element compartment that the pIMedia port is a port for MediaCtrl and there is no such relationship between the elements.
10				X		4.3	IAddMediaAlbum	An interface for adding media to the album has not been defined.
11				X		4.3	IManageMedia	No media management interface has been defined.