# SOFTWARE DESIGN

# HANDBOOK

# CONTENTS ...

# Project Glossary

> Project glossary defines the meaning of the vocabulary appearing in the software project. It ensures different users to maintain the same understanding when reviewing the software design artifacts.

## What is a project glossary?

Project glossary allows us to define a standard meaning of a particular vocabulary or phrases inside the software design artifacts, such as information captured in requirements, design diagrams, documents, etc… People with different background or perspective could have different understanding when they read on the same piece of information. Consider the following statement:

*Customer can remove his/her account permanently by mailing us an account removal form.*

Let's focus on the word **account** in the statement, what does it means? To a user with banking background he associated it as a bank account, but for a service provider this could mean the customer account for subscribing a particular service. This kind of ambiguity could causing serious communication issue between different parties involved in a software project, thus it could result a disaster as the incorrect interpretation of software design artifacts.

The project glossary is a collection of vocabularies or phrases (the terms) captured from various models, reports and any other artifacts in the software project. Each term are defined with its meaning specific to the project domain. By defining project glossary, it unifies the vocabularies from different sources into a standard meaning for different parties, that serves various different of purposes for project development:

- Understand the design artifacts
- Provide description to key terms and phrases,
- Explain detailed concepts
- Describe abbreviation, etc

## How to define project glossary?

Project glossary can be defined by reviewing and standardizing all the terms used from different models and artifacts in your project, including the requirements obtained from users, meeting notes, design specifications, test plan, etc… You can put all the identified terms into a list and then give them the meaning. Once complete you got a full list of terms used in your project. Everyone can refer to it when reviewing the project artifacts, which can avoids ambiguity causing by interpretations of a term from different stakeholders or using different expressions for the same terms.

| Name ▲ | Description |
| --- | --- |
| administrators | The person who owns the right to manage the upload and removal of |
| archive programs | To mark a program as archived. |
| archived TV programs | The program that was firstly broadcast for at least 3 months. |
| general member | The person who own the free membership, and is allowed to watch fre |
| monitor the delivery of newsletter | To make sure the newsletter are delivered. Failure delivery should be cl |
| monthly newsletter | An email to send to user every month, to give information on the com |
| OTV | A company which delivers both paid and free online broadcasting servi |
| premium member | The person who own the premium membership, and is allowed to watc |
| program schedule | The timetable of TV programs. |
| update the program | To update the list of programs on the programs page. |
| video panel | The panel for playing a program. |

## Link up similar terms with Alias

Sometimes multiple terms captured could meaning the same concept. In this case we can select one as the glossary, and have the rest defined as its alias. This help to make sure user can find the correct meaning to the term even it is different to the glossary being defined, without define the same meaning over and over on all the similar terms.
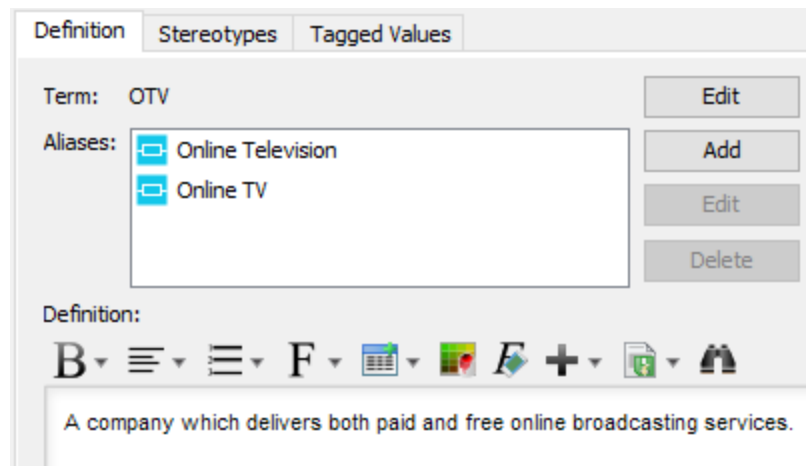


## Organize related terms together

For a complex project it could come up with thousands of glossary terms. Even if the terms are sorted in in order, it could be still very difficult and time consuming to looking up. We can organize the terms according its context that make terms easier to be looked up and located.

# Impact Analysis with Analysis Diagram

> Impact analysis is an important technique used in software development. It helps us to delimit the scope of impact while we make changes on the software system.  Oversee the risks of the change and the resources which we should plan forward for implementation of the change.

## What is Impact Analysis?

Impact analysis is a technique that helps to identify the potential consequences of any changes we would like to make on a software system. The world keeps changing; it is unavoidable that even the most well developed software system will need to make changes in order keep up with the challenging world. These changes may come from various sources, such as business needs adjustment, new requirements, new technologies, etc… But since software systems could be very complex, it is very likely that the changes could result in failure or becomes out of control.  It could be risky to jump into the software system directly making change without a proper plan. By conducting impact analysis we can learn what will be affected before making the actual changes. This helps us determine the scope of the change, along with estimation on the resources we need for planning the changes.

## How to Perform Impact Analysis?

An impact analysis involves 2 steps, first we have to find out the relationships related to the subject element, and then we review them to uncover any possible consequences and risks associated with the change. There are 2 ways to find out the relationships; by studying the static relationship as well as the dynamic linkage related to the subject element or modules.  We refer them as traceability study and dependency study.

On the traceability study, we review all static relationships related to the subject element, including requirements, specifications, design elements, test cases, etc…, which are defined in the specification and design of your software systems. On dependency study, we review all the logical and dynamic linkages related to the subject element including the use of variables, program logics, module architectures, etc...

After performing the traceability and dependency study we can obtain the full spectrum of relationships related to the subject elements. We can then move on to the second step, which walk through the relationships one by one for evaluating the impact for the related elements when making changes. A series of questions should be considered regarding the related elements being affected by the change, for example:

- Will the change lead to failure of running for other modules?
- Will the change causing performance drawback to the system?
- How much man power do we need to implement the change?

You may have more questions to ask during the review process depending on the subject of its nature, business decision as well as the complexity of your system.

# What is Analysis Diagram?

Nowadays many people are applying system analysis and design with CASE tools, we can perform impact analysis easily by creating an analysis diagram. The analysis diagram is a star shaped style chart with the subject being studied in the center. Surrounding with its related elements and relationships; identified from traceability study and dependency. Each related elements and relationships are labeled with its name and also its nature.
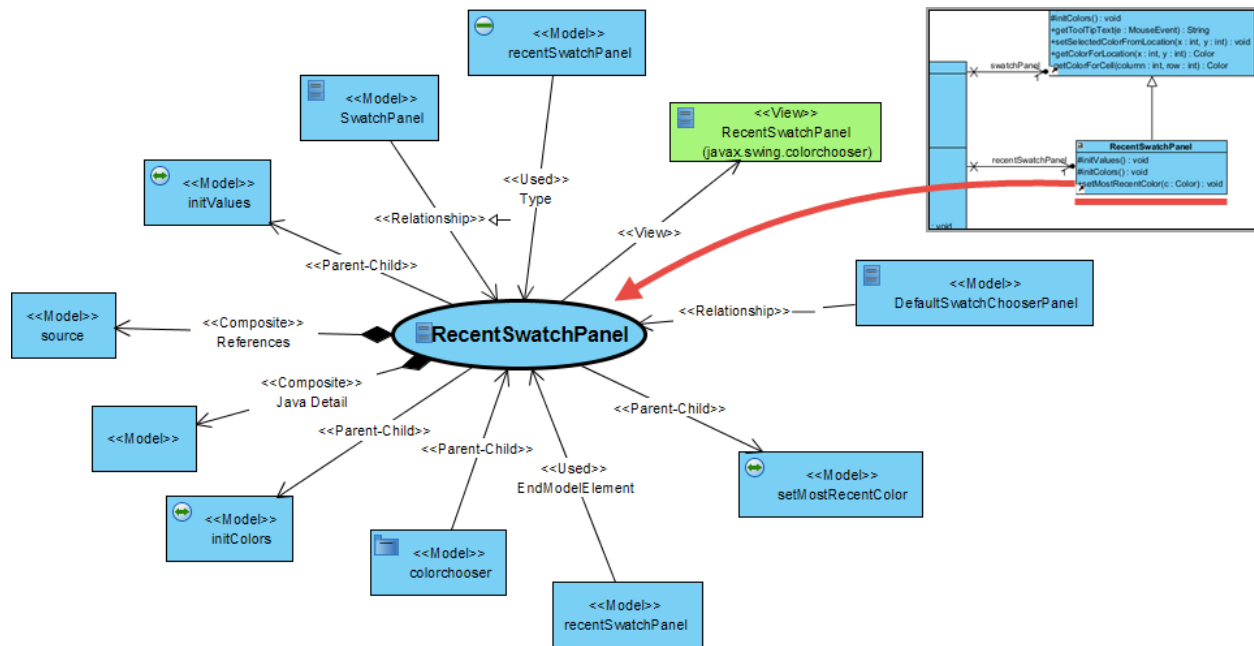


# Benefits of Analysis Diagram

## Easy to review

By visualizing all the scattered and unorganized relationships obtained from traceability and dependency study into a single graph, we can easily visualize and review the relationships among the related elements, and particularly how they are related to the subject element.

## Knowing the scope of the change

By reviewing the relationships on the analysis diagram we can learn which will be affected by the change and the level of impact it may be caused. For example if we are going to make change on a data field we can find out which user interface widget and database storage could be affected. This allows us to find out the scope of the change we will have to make.

## Determine the depth of change

The analysis diagram is not limit to a single level of relationship between the subject element and its related elements. For example, if we change the internal logic of an operation, while the class itself was used as a type by others, then the change could be risky.  Hence, it may cause the classes relying on it to become malfunctioned and we may need further investigate into the details on the class by which is using the subject element as a type. This will discover any further impact that may be imposed by our change.
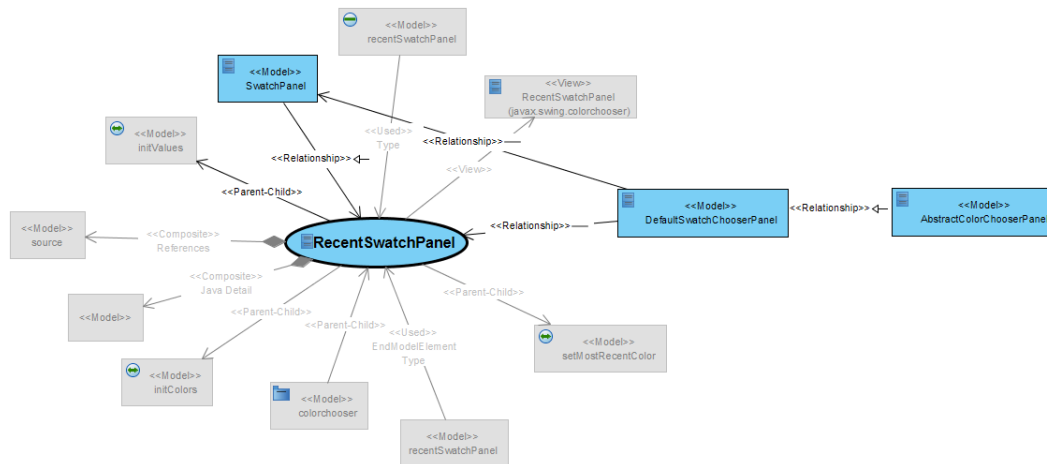
## Plan forward for the resources needed by the change

Having a clear understanding to the scope allows the team to make accurate estimation regarding the resources that maybe needed.   For instance duration of work, man power, etc... Thus the team can have a better planning for implementation of the change.

# Create in Ad-hoc

An analysis diagram may become obsolete as the system may have been changed from time to time. Therefore it should be created in ad-hoc basis. There is no point to keep the analysis diagram after you obtain the changed scope. We can simply discard it in order to keep the project simplified and light weighted. We can recreate an updated version whenever it is necessary.