Software Improvement Group

SIG/TÜVIT EVALUATION CRITERIA TRUSTED PRODUCT MAINTAINABILITY FOR VISUAL TECHNOLOGIES

ADDENDUM

SOFTWAREIMPROVEMENTGROUP.COM

Authors

Dennis Bijlsma
Julian Jansen
Pepijn van de Kamp
Rick Klompé
Xander Schrijen
Reinier Vis

Version: 1.0
April 17, 2020

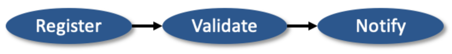
1. INTRODUCTION

This document is a companion to document SIG/TÜViT Evaluation Criteria Trusted Product Maintainability[1].

This guidance document provides an explanation to software producers about the measurement method SIG applies for evaluation in the context of ***Visual Technologies***. The goal of this document is to explain the relationship between the concepts that characterize Visual Technologies as described in the *ISO/IEC 19510 – Business Process Model and Annotation* to the concepts from the SIG Maintainability Model. Also, this document explains how the measurements used are performed on software artifacts related to Visual Technologies.

1.1 WHAT DOES SIG CONSIDER AS VISUAL TECHNOLOGIES?

Visual Technologies are the group of technologies where developers or technology users use a visual representation rather than a textual representation to express functionality of software.

Visual representation of functionality	Textual representation of functionality
Registration: 	`public class Registration{ public void register(){ validate(); persist(); notify(); } }`

```
public class Registration{
    public void register(){
        validate();
        persist();
        notify();
    }
}
```

SIG recognizes three main categories of Visual Technologies:

- **Low-code**
  Technologies that aim for rapid delivery of functionality by minimizing hand-coding.
  Example: Mendix, OutSystems, Microsoft Power Apps
- **System integration and Business Process Management (BPM)**
  Technologies that aim for modelling, governing and automating business processes.
  Example: IBM BPM, Oracle BPM, Tibco Business Works, Oracle BPEL, Apache Airflow
- **Data & BI pipelines**
  Technologies that aim for controlled processing of data in the large.
  Example: Oracle ETL, Informatica Powercenter, Microsoft SSIS, SAS

Note that it is not unusual for visual technologies to span multiple of the above categories.

[1] https://www.softwareimprovementgroup.com/methodologies/iso-iec-25010-2011-standard/

2. DEFINITIONS

This chapter describes the differences for Visual Technologies in the definitions of the concepts of System, Component, Module and Unit as described in *SIG/TÜViT Evaluation Criteria Trusted Product Maintainability*.

2.1 SYSTEM

No difference in the definition of a system compared to other technologies.

2.2 COMPONENT

No difference in the definition of a component compared to other technologies.

2.3 MODULE

The concept of a module in the SIG Maintainability can be referred to as a file or named artefact that contains one or multiple processes, graphs or flows. The ISO/IEC 19510 refers to the concept of a process, graph or flow as a Diagram. All diagrams that are defined in the same file are considered part of the same module in the SIG Maintainability Model.

The following table lists examples of technology-specific structures that are counted as modules in the SIG Maintainability Model:

Vendor	Module
Mendix	Microflow, Nanoflow
OutSystems	An OutSystems Process
Oracle BPEL	A .bpel file
Tibco	A .process file
Microsoft SSIS	A .dtsx file

While most technologies allow for specifying a single named diagram in a file, there may be exceptions where technologies allow for specifying multiple diagrams in a single file. In the case that multiple diagrams are specified in the same file, these diagrams are considered as being part of a single Module in the SIG Maintainability Model. This is similar to how in the SIG Maintainability Model a Java or C# file that contains multiple class definitions is considered as a single Module.

2.4 UNIT

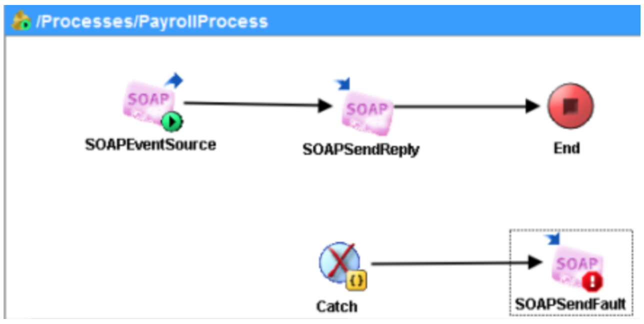
The notion of a Unit in the SIG Maintainability Model is defined as the smallest named piece of executable code. The ISO/IEC 19510 refers to the concept of a process, graph or flow as a Diagram as the smallest executable element. An artefact in a visual technology can consist of multiple diagrams. In such a case the artefact is the Module and each individual diagram is a Unit in the SIG Maintainability Model. Note that it is common for Visual

Technologies to have a single diagram specified per file. Hence, it is not uncommon for Modules to only consist of a single Unit.

Disconnected graphs in a single diagram

A unit consists of all the process elements that are part of the Process Diagram. Note that it is not relevant whether the process elements in the diagram are part of- or connected to- the same process graph. A unit may consist of multiple subgraphs that are not necessarily interconnected.

A common example of having isolated subgraphs within the same process diagram is when exception handling is implemented:



Example of a TIBCO process with Exception Handling

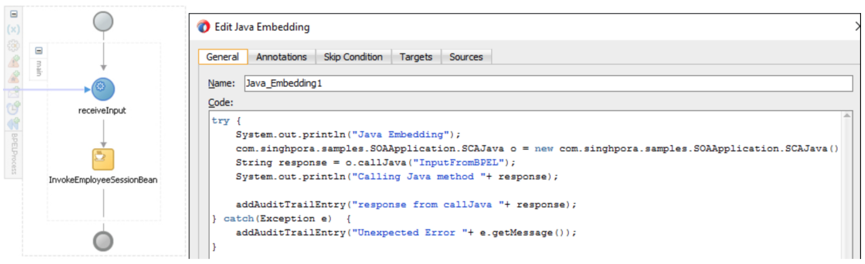
Sub-Processes

Sub-processes are counted as individual units as long as they are part of a separate diagram as the parent process. Typically, the parent process is able to access the sub-process using a Call Activity as described in the ISO/IEC 19510.

If a sub-process is part of the same process diagram as its parent or calling process, then it is also part of the same unit as the parent process.

Embedded code

Some Visual Technologies allow for defining embedded code inside the process element. With embedded code we mean textual code that is written in another technology than the visual technology itself.



Name: Java_Embedding1

Code:

```
try {
    System.out.println("Java Embedding");
    com.singhpora.samples.SOAApplication.SCAJava o = new com.singhpora.samples.SOAApplication.SCAJava()
    String response = o.callJava("InputFromBPEL");
    System.out.println("Calling Java method "+ response);

    addAuditTrailEntry("response from callJava "+ response);
} catch(Exception e)  {
    addAuditTrailEntry("Unexpected Error "+ e.getMessage());
}
```

Example of a BPEL process (left) with a Java Embedding activity that contains embedded code (right)

Each instance of embedded code is considered as its own unit and evaluated separately from the process it is part of. In the above example the BPEL process is measured against the Evaluation Criteria for Visual Technologies (this document) and the embedded code is measured as Java code against the regular SIG/TÜViT Evaluation Criteria. Common forms of embedded code in visual technologies are Java, C#, JavaScript and Xslt.

3. GUIDANCE FOR PRODUCERS

3.1 VOLUME

The volume of software artefacts written in visual technologies is not counted in terms of the number of lines of code in all files, but rather in terms of the number of process elements that are part of all process diagrams.

In the context of the SIG Maintainability Model, a *process element* is a node in a process graph that is involved in the process flow of a single process diagram.

The ISO/IEC 19510 describes the following process elements:

BPMN Element	Example	Counted for Volume	Explanation
Event	Start, Intermediate, End	Yes	Similar to method start and end tokens, like {, }, return.
Activity	Task, Call, Transaction	Yes	Similar to a line of code
Gateway	Condition, Loop	Yes	Similar to a control flow element, like *if, for, switch*, etc.
Flow	Sequence flow	No	Used to indicate process flow, not to influence it.
Group	Swimlane	No	Used to group process flow, not to influence it.
Artifact	Annotation	No	Can be compared to a code comment.
Data Object	File	No	Used in process flow, but not influencing it.

For each Visual Technology, a comparison of the process elements available in that technology to the process element types that are described in the ISO/IEC 19510 will be made to determine which process elements should be counted for the Volume property.

For the evaluation of the volume property, the software product's rebuild value is estimated on the basis of the number of process elements in production code. To make the number of process elements of software artefacts written in different Visual Technologies comparable to each other, they are normalized on the basis of industry averages.

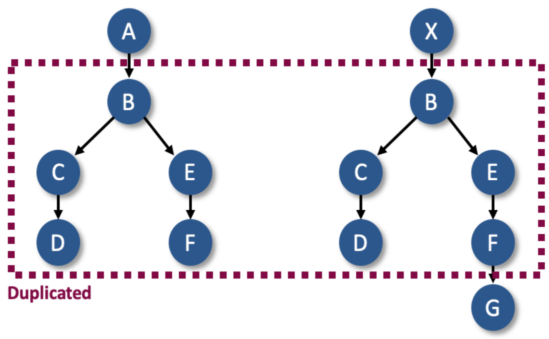
The following table lists the number of process elements that are produced on average for some industry visual technologies:

Language	Process Elements (26 person years)
TIBCO Business Works	8,000 process elements
Informatica Powercenter	52,000 process elements
OutSystems	156,000 process elements
Oracle BPM	8,000 process elements
WebMethods	32,000 process elements

SIG measures and collects productivity data for more than 300 technologies. Kindly contact us for more information on a specific technology.

3.2 DUPLICATION

For duplication, the number of process elements that occur in an identical structure of at least 3 process elements in other processes in the same system are measured. For each process element, the name, type, as well as incoming and outgoing edges are used to determine an identical structure.

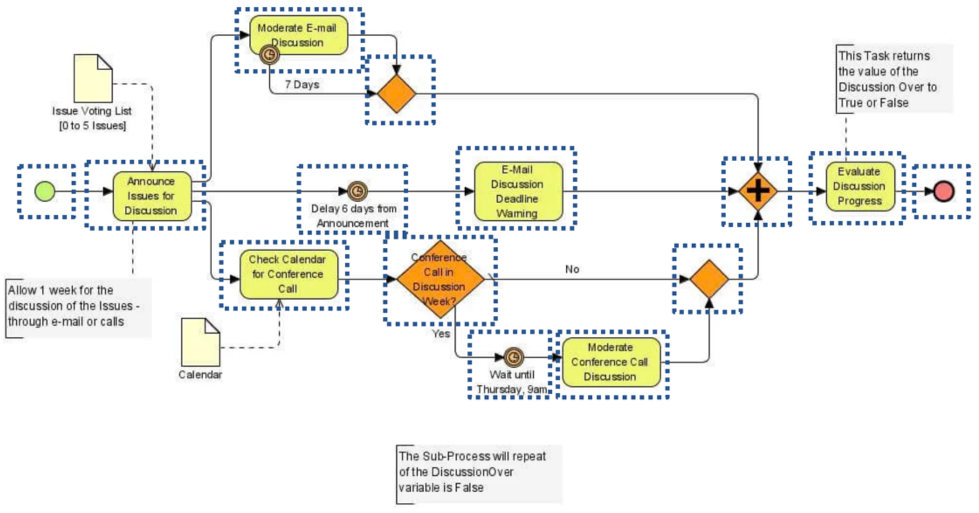


Example: Process elements B, C, D, E and F occur in the same structure both processes.
Out of the 13 process elements in both processes in the example, 5 process elements are redundant.

3.3 UNIT SIZE

Unit size is measured by determining the number of process elements per process. Only process elements that are counted for the Volume metric are considered. This means that process elements like Start events, End events, Activities and Gateways are all considered to contribute towards the size of the process.

Given the overview of process elements that are counted for the volume metric, the following example has 14 process elements.


Moderate E-mail Discussion
7 Days
Issue Voting List [0 to 5 Issues]
This Task returns the value of the Discussion Over to True or False
Announce Issues for Discussion
Delay 6 days from Announcement
E-Mail Discussion Deadline Warning
Evaluate Discussion Progress
Allow 1 week for the discussion of the Issues - through e-mail or calls
Check Calendar for Conference Call
Conference Call in Discussion Week?
No
Yes
Calendar
Wait until Thursday, 9am
Moderate Conference Call Discussion
The Sub-Process will repeat of the DiscussionOver variable is False

Example: A BPMN process with a process size of 14 process elements according to the SIG Maintainability Model

3.4 UNIT COMPLEXITY

Unit complexity is determined based on the McCabe Cyclomatic Complexity metric. For a process this metric can be calculated using the following formula:

McCabe Cyclomatic Complexity = **E – N + 2**
Where E is the total number of edges (arrows) and N is the total number of nodes (the process elements).

For example, the above BPMN process has a McCabe complexity of 5 (E=17, N=14, 17-14+2=5)

Gateway process elements that do not influence control flow but indicate parallel or sequential execution (e.g. parallel forks and join gateways) are not counted for unit complexity.

3.5 UNIT INTERFACING

No difference with unit interface measurements in other technologies.

Note that in process languages the input parameter is typically a single message or document.
For technologies that do not allow for defining an interface or input message for a process the unit interfacing system property is scored as Not Applicable.

3.6 MODULE COUPLING

No difference with module coupling measurements in other technologies.

3.7 COMPONENT BALANCE

No difference with component balance measurements in other technologies.

3.8 COMPONENT INDEPENDENCE

No difference with component independence measurements in other technologies.

3.9 COMPONENT ENTANGLEMENT

No difference with component entanglement measurements in other technologies.

Software Improvement Group

Fred. Roeskestraat 115
1076 EE Amsterdam
The Netherlands

www.softwareimprovementgroup.com
marketing@softwareimprovementgroup.com

Getting software right for a healthier digital world