VISDOM Visual software diagnostics

D3.2.1. Visualization guidelines for the different viewpoints.

Programme	ITEA3		
Challenge	Smart Engineering		
Project number	17038		
Project name	Visual diagnosis for DevOps software development		
Project duration	1st October 2018 – 30st June 2022		
Project website	https://iteavisdom.org/		
Project WP	WP*3* - Visualizations		
Project Task	Task *3.1 – Visualization viewpoints*		
Deliverable type	X	Doc	Textual deliverable
		SW	Software deliverable
Version	1.0		
Delivered	12/2020		
Access	X	Public	
		Abstracts are public	
		Confidential	

Version history

Version	Date	Author	Notes
0.1	26.10.2020	Kari Systä (TAU)	Sketching content to chapters 4 and 5
0.2	15.11.2020	Markus Kelanti (UoO), Henri Bomström (UoO)	Dashboard concept and stakeholder view
0.3	30.11.2020	Markus Kelanti (UoO)	General rework of the document
0.4	08.12.2020	Kari Systä (TAU)	Commenting/editing round
0.5	10.12.2020	Henri Bomström (UoU)	Rewrote chapter 3.
0.6	15.12.2020	Kari Systä (TAU)	Review, minor changes and comments,
0.7	17.12.2020	Markus Kelanti (UoO)	Revising chapter 3, 4 and 5.

0.7	18.12.2020	Henri Bomström (UoU)	Added definitions for visualisations and dashboards. Minor other edits.
0.8	18.12.2020	Henri Bomström (UoU)	Added links to source code repositories.
1.0	31.12.2020	Markus Kelanti (UoO)	Prepare v. 1.0

1 Introduction

This document describes general guidelines for creating configurable visualisations and dashboards supporting generation of stakeholder specific views on software process.

Purpose of the guidelines is not strictly define how different dashboards and visualisations should work but rather provide a general guidelines and information what should be considered when creating visualisations that are capable of changing based on different needs and constructing views on data created from different visualisations or groups of visualisations.

For the reader, this document contains guidelines for:

- creating visualisations that support stakeholder viewpoints (Chapter 2)
- creating a configurable dashboard supporting different stakeholder viewpoint generation (Chapter 3)
- general styles of visualisations and views in VISDOM -project (Chapter 4)

Finally, appendix A gives recommendations and guidelines for creating a graphical design for individual visualisations.

2 Stakeholder viewpoint design guidelines

The purpose of creating stakeholder specific visualisations is to allow stakeholders in different roles in a software development process to:

- see only the information that is relevant for the stakeholder
- communicate the information in the most efficient format for the stakeholder

The guidelines for stakeholder specific visualisations are based on a genre -based communication structuring[1]. Genre-based analysis of communication is a useful tool to understand the structure of communication practices in communities. The approach generally describes a document -based approach to describe the communication itself, creating a suitable tool to be utilised in VISDOM - based approach of stakeholder viewpoints. Essentially, this creates a framework that allows to structure and identify the information required to show different viewpoints on the same data and utilise different visualisations in different viewpoints. This approach allows us to structure individual visualisations or dashboards implementing one or multiple visualisations or other dashboards as part of communication in a software development process. The following guidelines for creating visualisations to support stakeholder viewpoints were created during the project.

Guideline: *When designing visualisations and dashboards, the stakeholder, intent and context should be identified.*

The information about who (stakeholder) uses the visualisation, for what purpose (intent), and in what information is required (context) is identified in order to create a visualisation that supports the stakeholder:

- **Stakeholder.** Stakeholder in this context represents any actor (including human and machine actors) that has some intent, viewpoint and context that describes why, how and in what presentation format should be used to view any piece of information.
- **Intent.** Intent is a description of what the stakeholder wants to do with the information. It allows identification of what kind of information is needed in order to satisfy the intent.
- **Context.** Defines what information is required to shown in the viewpoint from all available data.

[1] Orlikowski, W., & Yates, J. (1994). Genre Repertoire: The Structuring of Communicative Practices in Organizations. *Administrative Science Quarterly, 39*(4), 541-574. doi:10.2307/2393771

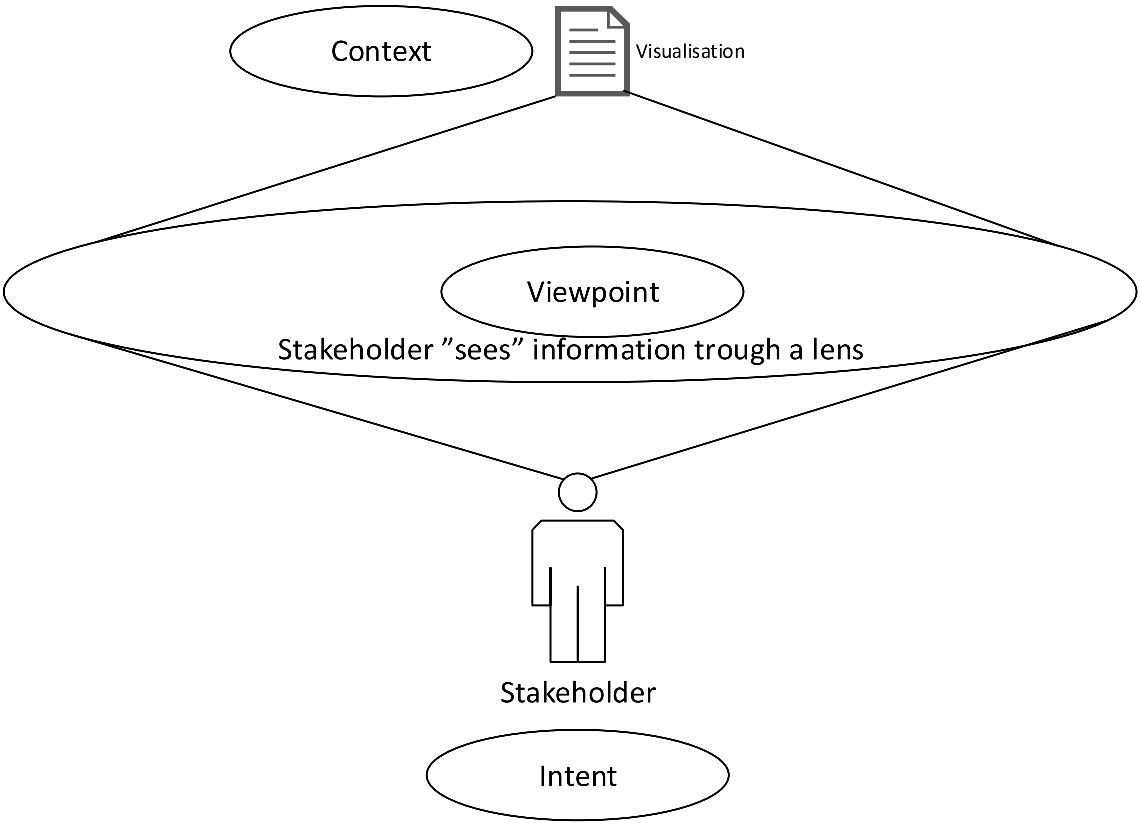

Figure 1. Stakeholder viewpoint.

The purpose of the information presented before and in Figure 1 is to understand and describe the viewpoint a stakeholder has to the information. It should be noted that the intent and context aren’t exactly defined, they represent the information required to guide the selection process, allowing the dashboard to select suitable visualisations.

Guideline: *Viewpoint represents how the stakeholder prefers to see information.*

The viewpoint concept in VISDOM is used to describe what the stakeholder wants to see from the individual stakeholder perspective. It provides information that how the information should be represented for the stakeholder so it can be utilised. This includes visual and machine representations/data formats.

The viewpoint is used as a way to identify individual visualisation(s) that supports certain kind of stakeholder needs. The viewpoint is used in the configurable dashboard (presented in chapter 3) as a descriptor that is used to construct a concrete dashboard. Figure 2 provides an example of two visualisation services providing the necessary visualisations to support the viewpoint.

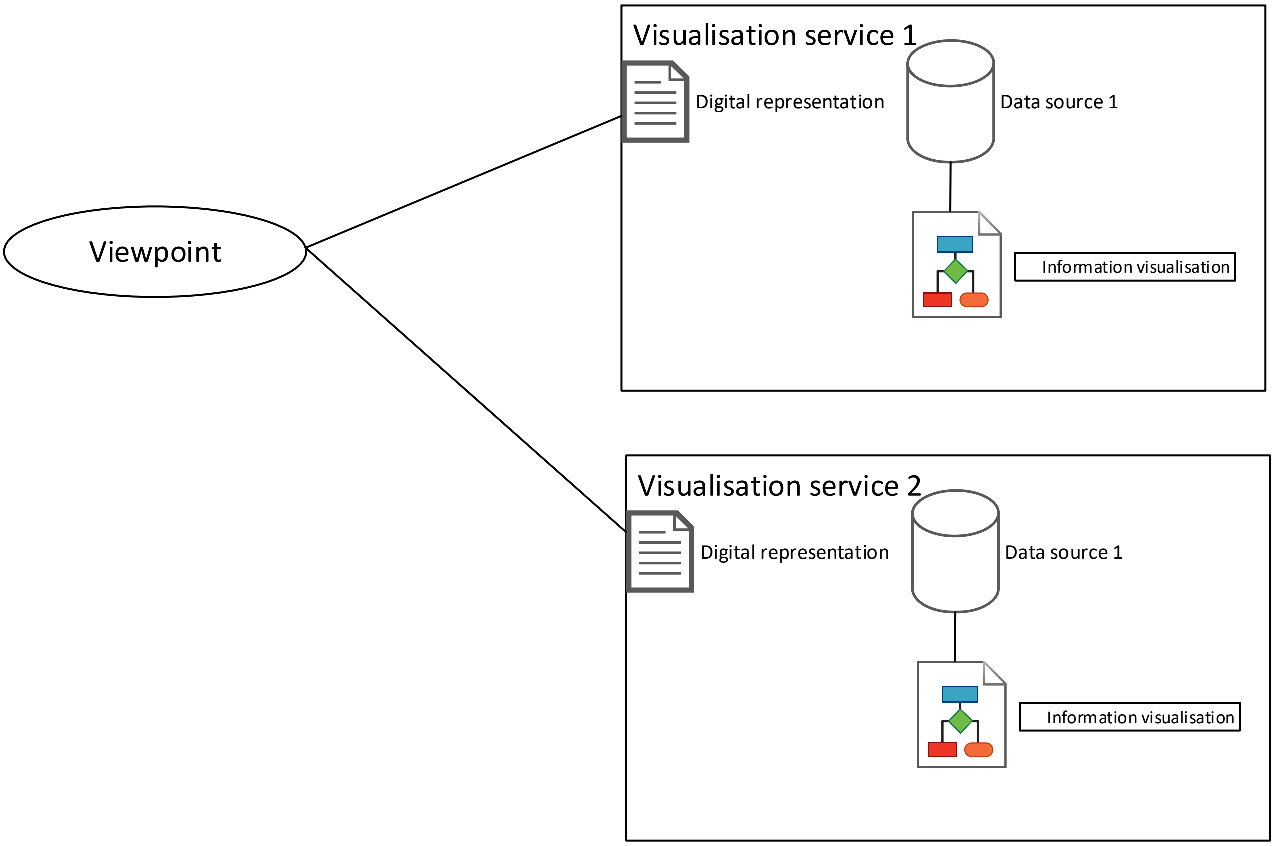

Figure 2. Example of a viewpoint contains two visualisations that create the view needed by a stakeholder.

Guideline: *Each individual visualisation should have a digital representation.*

In order to generate viewpoints in a dashboard, both the visualisation and dashboard itself should have a digital representation, such as digital twin, that contains information about what stakeholder uses such visualisation, what are the intents it supports, what context it supports and what viewpoints it support. The actual implementation depends on how the configurable dashboard is implemented but the digital representation allows the dashboard to identify proper visualisations and communicate with them.

Guideline: *Visualisation dashboard should support viewing individual visualisations and existing viewpoints.*

Dashboard is not necessary just a collection of different visualisations supporting certain stakeholder needs but should also be able to contain viewpoints in addition to individual visualisations. This is required as certain view can support other view when added with additional visualisations or other views.

Figure 3 provides an example of a view that contains 2 individual visualisations and one existing viewpoint creating a new viewpoint.

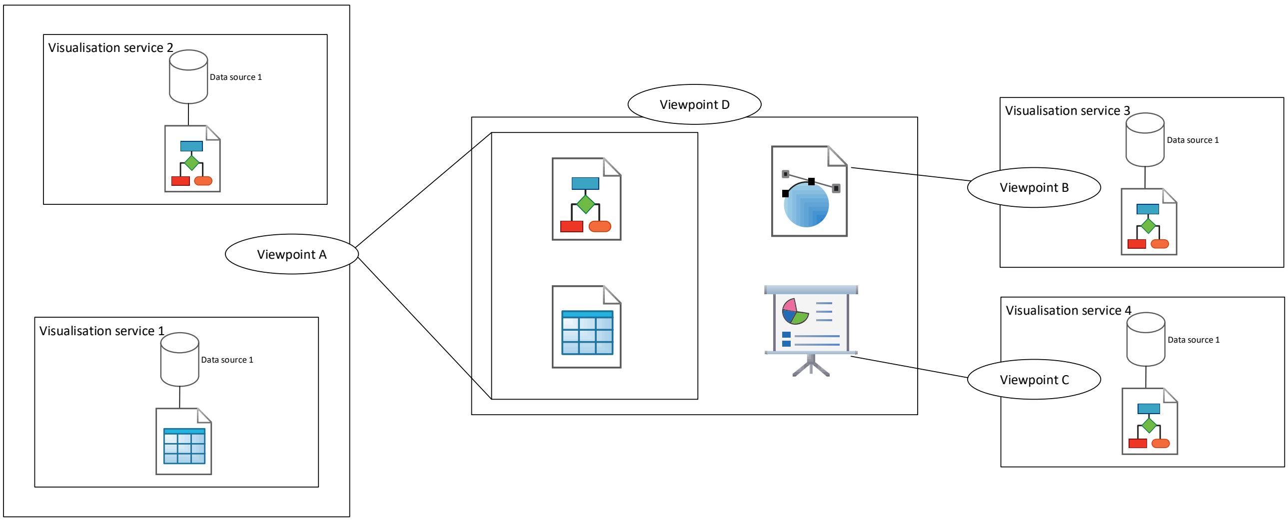

Figure 3 Example of a viewpoint containing 2 individual views and an existing viewpoint.

3 Configurable dashboard design guidelines

Delivarable D2.5.1 defines the overall VISDOM architecture. This section describes the dashboard related parts of the proposed architecture (Figure 4) and provides guidelines for implementing it for each specific case.

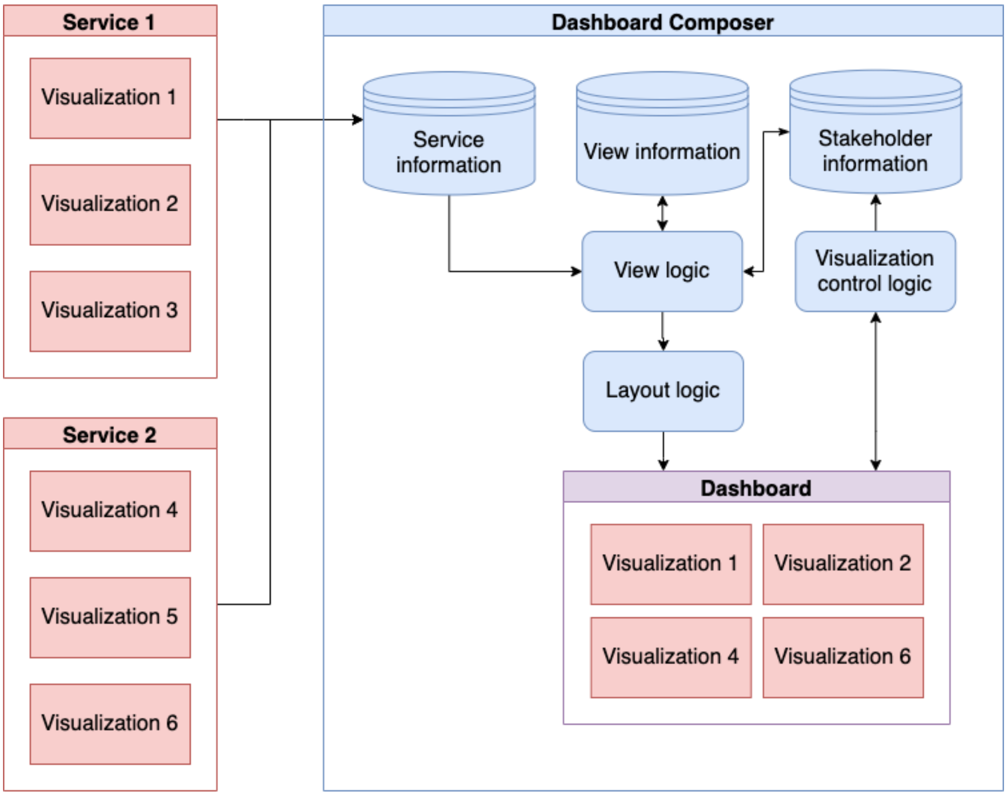

Figure 4. The dashboard related architecture comprises of visualization services, composition logic and of the dashboard view itself.

3.1 Visualisations

The configurable dashboard treats visualisations as independent software solutions that are imported to the dashboard. From a dashboard perspective, visualisations and their operational principles are seen as black boxes. Each visualization is built on a technical blueprint provided with the dashboard composer, that allows them to be integrated to the shared infrastructure. Microfrontends or plugin-type solutions are examples of individual software components that can be integrated thanks to a common interface. There are no further strict guidelines for technical solutions, as long as they do not interfere with dashboard integration. Multiple visualisations can be operated as stand-alone solutions or grouped into services that share visual attributes and authentication and authorisation schemes.

The provided blueprint features two main integration functionalities; visualisations must communicate their purpose and capabilities to the dashboard, and they can implement an interaction-oriented interface. The first functionality must enforce that visualisations communicate their purpose and capabilities to ensure seamless integration to the dashboard. The second feature allows them to communicate with each other's and receive instructions via the dashboard. The dashboard does not mandate, but does highly encourage, that visualizations implement the second specification and final the decision is left for the implementor. Communication is encouraged for uniformity between visualization implementations to coordinate colour schemes, sizes and shapes, topics of interest and changes in visualization perspective.

The following guidelines are given for implementing visualizations that can be integrated to the VISDOM dashboard solution:

- Visualisation implementations should be created from a common blueprint that integrates to the shared infrastructure in a plug-and-play style.
- The visualisation blueprint must enforce the visualisation to communicate its purpose and capabilities in a way that is compatible with the genre-based approach.
- The visualisation blueprint should offer an interface for inter-visualisation and dashboard composer related communication but should not mandate its use. Visualizations should strive for implementing the interface to the best of their abilities.
- Implemented visualisations must handle their own data retrieval, processing, and presentation.
- Visualisations should be responsive to different presentation and layout formats.
- When implementing multiple visualisations for similar purposes, they should each implement similar authentication and authorisation schemes, and strive for visual cohesion with similar colours, symbols and other visual means. Visualisations may be both bundled into services or implemented as stand-alone solutions.

An example of a configurable dashboard is available in the VISDOM website [2] where users can add and change different visualisations based on their need.

3.2 The dashboard composer

The dashboard composer is a software component responsible for the underlying dashboard logic. The composer provides the means for importing visualisations to the system and is responsible for communication and layout. Primarily, visualizations and the composer communicate via events. These events contain information related to the actions and configurations needed to achieve a uniform view from heterogenous visualizations. Additionally, the composer handles layout for the dashboard. To this end, visualisations must provide metadata about their purpose and capabilities, which the composer utilises in layout. Visualisations may request specific layout options, but the composer is responsible for making the final decision.

The VISDOM architecture supports two main technical approaches for relaying events on a configurable dashboard. The first approach relies on the browser to act as an event-based communication channel between visualizations, whereas the second one assumes that each

[2] https://iteavisdom.org/dashboard/

visualization connects to a message broker. The solutions do not exclude each other's but it may be best to adopt the most simple and consistent solution for each case. In the first option, the browser-based event system, visualizations subscribe to events distributed through the browser itself. This approach is relatively simple and requires no external tools. However, the downsides include difficulties for browser compatibility and challenges in centralized control of events. The second option relies on each visualization connecting to a shared message broker. This approach requires more development effort and may introduce problems related to distributed systems. However, it may offer unique benefits, when compared to a purely browser-based solution, such as better access control, external control over visualizations, and an option for gathering telemetry information. The browser-based approach represents a more simple and self-contained approach for implementing visualizations. To contrast, the broker-based approach allows for a more mature, controllable, and production-ready environment.

Guidelines

- The dashboard composer must allow visualisations to be imported to the system in an accessible manner and retain a list of imported visualisations.
- Individual visualisations are be treated as black boxes that are utilised based on the metadata describing their purpose and capabilities.
- The composer decides and orchestrates the final layout and presentation for each view. Visualisations may request specific layout and colour scheme related options, but the final say is left to the composer.
- The composer must be able to orchestrate visualisations to form a coherent view. Orchestration includes changing visualisation focus, colour scheme, and other visual attributes for each active visualisation to facilitate interaction between visualisations and to create a uniform look.

3.3 Dashboard configurability

The dashboard as a whole must be configurable for each use case and user. As a starting point, the dashboard should provide a default role, or other suitable attribute, -based dashboard out-of-the-box. These attributes must be configurable for each use case and may be bound to a custom authentication and authorisation scheme. For further usage, the dashboard must allow its users to create customized views that support their workflow. The users should be able describe what they are attempting to do, and the dashboard should try to match available solutions to their need. Eventually, the user decides whether to accept suggestions or to build a completely custom view. The dashboard should, if acceptable, feature anonymous analytics that describes how it is used as time progresses. This data should be used to target recommendations to specific tasks and to improve the default dashboards.

Guidelines:

- The dashboard should provide a default role, or other suitable attribute, -based dashboard out-of-the-box. The roles, and other suitable factors, should be configurable for each dashboard instance.
- The dashboard should allow users to create their own views from the available visualisations and layouts.

- The dashboard should, if possible, gather anonymous analytics from visualisation usage to improve default dashboard composition.

4 High-level visualization styles and guidelines

FPP and deliverable D3.1.1. (Designs of the new visualizations) defines four health-check-inspired visualizations: EKG, Pulse, Blood Pressure and Pulse. In this document we provide some guidelines for their design and implementation.

In the following guidelines the term *indicator* is used for derived (or sometimes raw) data that need to be communicated with the visualization.

4.1 EKG

The FPP document describes EKG as follows: *"The **EKG** of a software project will show the rhythm of pace of a project. Software projects typically include various repeating tasks and iterations, such as sprints or delivery cycles. Such phenomena can be shown as repeating visual patterns, similar to the rises and falls of an EKG curve. Problems will be quick and easy to spot as variations in the patterns and especially difference between planned and real state.*

According to D3.1.1. characteristics of EKG visualization for software projects are:

- Visualization consists of data from several sources, e.g., issue tracking, version management, testing tool, CI/CD pipeline.
- Visualization is composed of repeating shapes representing cyclic activities like frequent deployments.
- Visual appearance of normal and unusual situation is different.
- The visualized deviations represent some interesting phenomenon to some stakeholders.

From these we derive the following guidelines:

- For each EKG visualization a set of key indicators need to defined first. The assumed sensible size for the set is between 3-7. For less than 3 indicators we assume that some other visualizations are more efficient, and more than 7 indicators make the visualization more complex. (Reference: the medical EKG is based on 5 indicators).
- Since effective EKG visualizations are difficult to design, an iterative process is needed. This means that the implementation architecture support fast changes of the mapping of indicators to visual variables.
- A natural layout of EKG is based on timelines with zooming and panning support.

4.2 Pulse

The FPP document describes Pulse as follows: *"**Pulse(s)** are used to visualize the different paces of different parts of the project (process). Visualizing the pulses, i.e., paces, of sub processes within a project can help notice bottlenecks or other local problems that can harm the project later."*

The following guidelines are set:

- Pulse is a good visualization for repeating timing-related indicators, e.g., frequency/rate of something.
- Pulse is based on a single indicator that may be based either on single data, or on more complex combinations of several data items.
- The indicators may have upper and lower thresholds and breaking of these thresholds should be clearly visualization.
- Visualizations can be either displays of current values or trends in a timeline.

4.3 Blood Pressure

The FPP document describes blood pressure as follows: *"**Blood pressure** indicates the stress level of a project - for example, an unusually high level of activities in proportion to how much the project is progressing or a sudden increase in the number of contributors would show as high blood pressure. A rise in the project's blood pressure will be a trigger to make changes and react early enough."*

The following guidelines are set:

- The indicator is either a symptom of pressure, e.g., stress-level of developers or degraded quality, or an indicator whose correlation to negative effects is known in advance.
- The indicators may have upper and lower thresholds and breaking of these thresholds should be clearly visualization.
- Visualizations can be either displays of current values or trends in a timeline.

4.4 X-Ray

The FPP document describes the X-ray as follows: *"**X-ray** is needed to actually discover the "broken bones" of the project - code, architectural artefacts, and processes. While the aforementioned visualizations will give stakeholders instant feedback on the state of the project, in order to locate the cause of a problem when one arises, a detailed yet understandable view of the software and processes is required."*

Unlike the previous visualizations, X-Ray do not assume any numeric information. Rather it is for X-Ray is a mechanism to zoom in and out of the system or process. The other visualizations can be used as a component, or X-Ray can be a mechanism to link visualizations together for interactive exploration.

Appendix A: Semiology of graphics

The quality of visualization is based in the following criteria

- **Semiotic Clarity**: correspondence between semantic constructors and graphical symbols.
- **Perceptual Discriminability**: symbols distinguishable from each other.
- **Semantic Transparency**: meaning of a symbol can be inferred from its appearance.
- **Complexity Management**: present information without overloading the human mind
- **Cognitive Integration**: when multiple diagrams are used
- **Visual Expressiveness**: e.g number of visual variables used in a notation
- **Dual Coding**: e.g., use of text and graphics together
- **Graphic Economy**: the number of different graphical symbols should be cognitively manageable.
- **Cognitive Fit**: suitable for different tasks and different audiences.

And we will give guidelines based on these criteria.

NOTE: this section is still too much copy-paste from Hugo's thesis and other sources.

4.5 Semiotic clarity

The semiotic clarity discussed the mapping between ontological concepts (S) and visualization symbols (S). Ideally there mapping

one Symbol ⟺ one Concept

but there are often problems like:

- Symbol Redundancy: many S ⟺ one C
- Symbol Overload: one S ⟺ many C
- Symbol Excess: one or many S ⟺ zero C
- Symbol Deficit: zero S ⟺ one or many C

The problems are exemplified in Figure 5

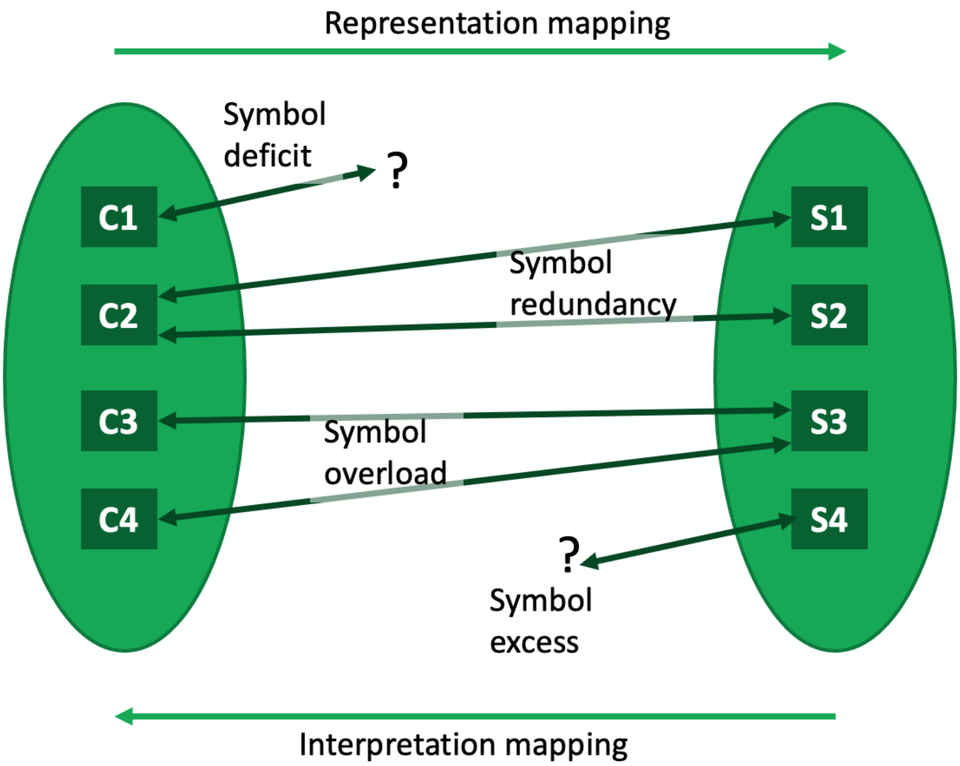

Figure 5 Problems in semiotic clarity

In the VISDOM visualizations we should avoid these problems and use the ideal mapping as an evaluation criterion for each visualization.

4.6 Perceptual Discriminability

This quality means that symbols in visualizations are distinguishable from each other. In practice this means that the Visual Distance between visualization symbols is big enough. A central tool to implement visual distance is proper use of visual variables. Typical visual variables are shown in below in figure 6.

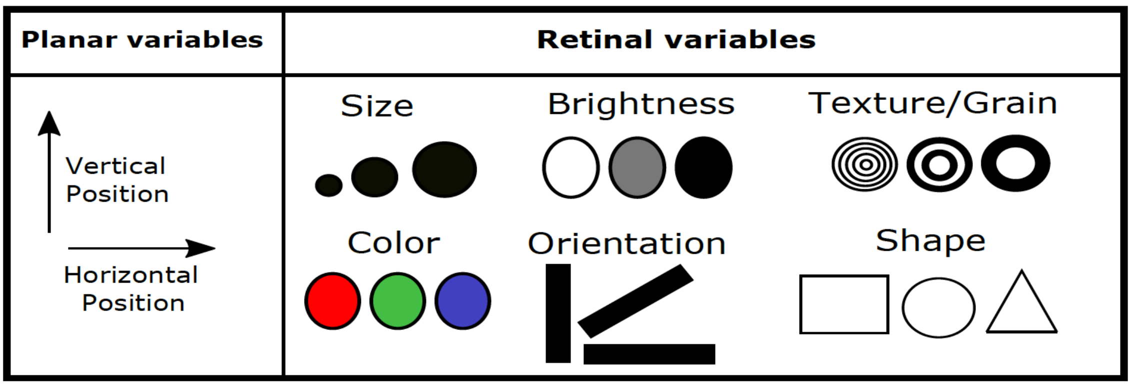

Figure 6 Visual variables affecting visual distance [3].

The variables differ in their qualities

- Only size is quantitative
- Size, brightness and sometime grain is ordered
- Etc ..

Similarly, number of distinguishable values vary. For instance, the research says that the capacity of retinal variables for a few shapes works as follows (see footnote 3):

	Size	Brightness	Grain	Color	Orientation
Points	4	3	2 to 3	7	4
Lines	4	4	3 to 4	7	2
Zones	5	5	4 to 5	7	-

Often it is a good idea to increase redundancy, e.g., use several variables to build enough distance.

4.7 Semantic Transparency

Graphical symbols should provide cues to their meaning. This means that the symbols should be (quote from Hugo's thesis)

- **Semantically immediate** symbols can be directly understood even by a novice simply through its appearance.
- **Semantically opaque** symbols have a purely arbitrary relationship between the symbol and its meaning.
- **Semantically perverse** symbols can induce a novice to infer a different or opposite meaning based on the appearance of the symbol.
- **Semantically transparent** symbols provide a cue to the meaning, but require some initial explanation.

These rules should be applied to all graphical symbols, including icons and relationships.

In VISDOM we have the special challenge to support different stakeholders with different stakeholders. The visualizations should be transparent to each stakeholder.

4.8 Complexity Management

We should include explicit mechanisms to manage complexity. These mechanisms should tackle human limitations like

- Perceptual: the ability to discriminate between elements
- Cognitive: capacity of working-memory

[3] Yosser El Ahmar, Enhancing the Cognitive Effectiveness of UML Diagrams: Application of the Semiology of Graphics. Doctoral Thesis, University of Lille, 2018.

Modularization is an efficient tool to manage complexity – large visualization should be divided in sub-part. One special way to implement modularity is to use hierarchy with top-down decomposition or levels of abstraction.

4.9 Cognitive Integration

Cognitive integration means that symbols in visualizations and in their components integrate is a correct way:

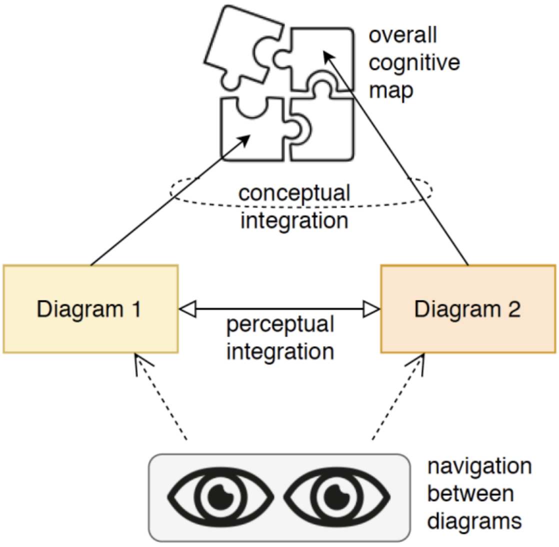

Figure 7 Cognitive integration[4].

Cognitive interaction of several visualizations is always important, and because VISDOM aims dashboards that contain several visualizations.

4.10 Visual Expressiveness

Visual expressiveness is closely related to Perceptual Discriminability and demands effective use of visual variables. "While the perceptual discriminability describes the visual variations between two graphical symbols, visual expressiveness measures the utilization of the graphic design space. Information-carrying variables are variables used to encode information, and free variables are variables that are not formally used in the notation. The degree of visual freedom is the number of free variables used, and is the inverse of the visual expressiveness."[4]

"While color is one of the most effective visual variables, it should not be used alone but in conjunction with others. The choice of the variables used should be based on the nature of the information: form follows content. The aim is to match the properties of the visual variable with those of the construct to be represented."[4]

[4] Fooy, H.: Software usage data visualization. Master's thesis, Tampere University (2019)

4.11 Dual Coding: e.g., use of text and graphics together

Text is not super efficient in visualizations but is a good complement.

Note that there are also textual annotations that provide additional information instead of dual coding.

4.12 Graphic Economy

We should aim at simplicity by minimizing the number graphical symbols. Naturally, we should still avoid symbol deficit (see Section 5.1). The lower number of symbols can be compensated with richer use of visual variables.

Also, all the means to manage complexity contribute to graphic economy, too.

4.13 Cognitive Fit

The visualization should fit the experience and background of the users. This includes both the experience level and expertise area.

For VISDOM this means that used symbols, visual variables and ways to manage complexity may vary for different stake holders (see Chapter 2).