On Requirements Visualization

Orlena C.Z. Gotel, Francis T. Marchese
Department of Computer Science
Pace University, New York, USA
ogotel@pace.edu, fmarchese@pace.edu

Stephen J. Morris
Department of Computing
City University, London, UK
sjm@soi.city.ac.uk

Abstract

This paper summarizes the typical objectives and process of visualization and highlights the primary areas in which visualization systems and artifacts have been used to support requirements engineering activities to date. The paper suggests that the field has yet to realize some of the benefits that can arise from a well designed and task-oriented information visualization, falling behind other areas of software engineering in which visualization has been used to better effect. By way of an exemplar, the paper proposes the need for a way to visualize the multi-dimensional nature of requirements to help bring about a shared and rapid comprehension on the health of a project's requirements, and so support various diagnostic activities and decision making tasks during software development. It examines how new ways to 'see' the requirements could be developed, based on metaphor and mapping, provides some samples, and outlines a research agenda to explore a vision related to requirements sensing.

1. Introduction

Effective visualizations are designed artifacts. As with all designed artifacts, there are stakeholders and end goals that these visual artifacts are intended to satisfy, along with defined processes that can help to achieve these ends. This paper questions the limited use of visualization in supporting requirements engineering activities to date, as compared with other areas of software engineering which appear to have received more attention and been the subject of international workshops and symposia in the area (e.g., the ACM Symposium on Software Visualization series, running since 2001, and the IEEE International Workshop on Visualizing Software for Understanding and Analysis series, running since 2002). There is undoubtedly much scope to advance the state of both research and practice in requirements engineering visualization.

This paper outlines a typical requirements comprehension problem that we suggest a carefully designed visualization could assist with, that of gaining a quick assessment on the 'health' of a set of requirements, a task that can be impeded by the need to browse through disjoint textual requirements documentation and accompanying models. It is common practice within other domains to reduce vast amounts of multi-dimensional data to a single picture to promote rapid situational awareness and enable decision making tasks, as with the military Common Operational Picture (COP) [44]. Within the software domain, the shared physical story wall that is at the heart of agile software development processes is the nearest approximation to such a COP. This affords project stakeholders with visual information about the changing status of stories (aka requirements) along with the option of physical manipulation. Virtual variants of such walls are now commonly developed and used [13, 30]. This paper builds upon the idea of shared visual communicative artifacts and sketches an initial concept for producing requirements pictures.

The near term vision is of a system that automatically maps data about requirements into visual artifacts, permitting stakeholders to actually 'see' the requirements, gain awareness on requirements properties and support high-level decision making activities. The longer term vision is to provide stakeholders with a way to 'sense' the essential characteristics of these requirements in a more direct and engaging manner.

This paper is organized as follows. Section 2 provides some background to the theory and process of visualization. Section 3 gives a brief synopsis of the use of visualization in software and requirements engineering. Section 4 describes how an effectively designed visualization could provide assistance in understanding various properties about requirements in support of typical stakeholder decision making tasks. Section 5 outlines how a visual metaphor and transformation system for generating requirements visualizations could be developed. Section 6 presents three sample visualizations. It explains their scope and intention, along with the generation procedure. These point to the infinite varieties of visualization that are possible in this domain, so a research agenda to pursue and validate this line of work is given in Section 7. The paper is presented to stimulate workshop discussion.

2. Visualization

Visualization is defined as *"the act of forming a mental vision, image, or picture of (something not visible or present to the sight, or of an abstraction); to make visible to the mind or imagination"* [33]. Visualization is also a form of computing, the goal of which is to arouse consciousness and insight. It transforms data for easier assimilation by an individual's sense of sight. Visualization algorithms restructure numerical and symbolic data in visually perceived forms. This means that visualization must be concerned with those mechanisms within humans and computers that allow the perception, use and communication of sensory information. As such, visualization draws upon many fields for its foundations, including: computer graphics, computer vision, computer science, human computer interaction, art and design, cognitive science and artificial intelligence. In computer supported visualization, complex data is mapped to perceptual representations in such a way as to maximize human understanding and communication. Therefore, the goal of the computer visualization process is to engender a deeper understanding of information, physical phenomena or the underlying processes related to them.

The visualization process, in its simplest form, is a sequence of steps that include the gathering, processing, pictorial rendering, analyzing and interpreting of data. This is the traditional data flow model of visualization. More formally, each step in the visualization process requires the design and transformation of a model. For example, in the scientific visualization process outlined by Earnshaw, there are three modeling steps [15]. The first modeling step takes place at the beginning of the visualization process in which a physical model is created that defines how the real world is to be viewed. This conceptual model sets a framework for the design of experiments and interpretation of data derived from them. In the second modeling step, the conceptual model is transformed into a formal mathematical model. Such a model can be subjected to rigorous transformation, analysis and proof. In the final step, the mathematical model is transformed into an approximation that is solvable by computer. The result is a simulation whose output is evaluated to test modeling assumptions, assess which physical phenomena are present in the data, and determine what improvements are required to the physical and mathematical models.

A clear correspondence exists between visualization and software engineering processes in which domain, design, and implementation models match up precisely with their scientific analogs. Moreover, the essential question asked about a computer simulation and a software system is the same – how well does the implementation model embody the underlying conceptual representation?

The key step in visualization is the transformation of data into a graphical representation. Haber and McNabb have proposed a model in which this is carried out by a sequence of three classes of transformations [20]. First, data enrichment or enhancement operators process raw data through numerical analysis or image processing techniques such as interpolation or filtering. Second, visualization mapping constructs an imaginary object (an imaginary object with some extent in space and time) called an abstract visualization object (AVO) and maps data onto attribute fields of an AVO. Fields include geometry, color, tint, reflectance, surface texture and others. Transfer functions define the mapping from raw data to AVO and take many forms. The simplest is a linear mapping that preserves quantitative information. The third transform is rendering. The rendering transform operates on the AVO to produce a displayable image.

How an AVO appears depends on the domain in which the visualization is performed. For example, in scientific visualization, representations of natural phenomenon as numeric data and mathematical models are manipulated to bring more insight to the phenomenon. There is typically a one-to-one mapping between a computer generated image and the underlying conceptual model leading to visualizations that attempt to render faithfully that model. So an AVO, such as a ball-and-stick molecular model or topographic relief map, will be a 'faithful' realization, rendering or reification of the underlying conceptual model, built to represent symbolically a particular domain and to be testable with data from that domain. In contrast, information visualizations typically render abstract data that is not necessarily linked to a physical substrate (e.g., distribution of library books by call number, web searches by age and gender, location of nucleotide sequences in the genome, etc.) [41]. In these cases, information visualization designers rely on creating AVOs that directly connect with the viewer's visual perceptual skills by judiciously selecting and integrating color, shape and texture to create a Gestalt, a unified arrangement of elements that convey a coherent message.

Fundamental to AVO design is the ability of a viewer to construct a mental model, the visual attributes of which represent data in a definable way. Therefore, the problem facing a visualization designer is the creation or selection of an appropriate representation or sets of representations that, according to Robertson, *"can provide the key to critical and comprehensive appreciation of the data, thus benefiting subsequent analysis, processing, or decision making"* [35]. As such, the designer must answer several questions:

- What mental models most effectively carry various kinds of information?
- Which definable and recognizable visual attributes of these models are most useful for conveying specific information either independently or in conjunction with other attributes?
- How can we most effectively induce chosen mental models in the mind of an observer?
- How can we provide guidance on choosing appropriate models and their attributes to a human or automated display designer?

This is a particularly challenging problem for representing complex knowledge, such as that created during a software development lifecycle, that must integrate information with *"insights, experiences, attitudes, values, expectations, perspectives, opinions and predictions"* [17]. It is usually solved by invoking a cartographic paradigm [16], but the field remains open for innovation.

Moreover, there is a major jump from 'data' to 'information' involving, however one likes to define it, some significant changes in semantics, purpose and use. Associated with this transition is a fundamental change in the range of visual symbols used for representation, which may become purely arbitrary, at least until conventions become established. At the extreme position of generalization, or subjectivity, are the visual forms used to represent *ad hoc* 'conceptual models' generated within particular domains to hypothesize new informational entities and the relationships between them.

3. Requirements Engineering Visualization

Information visualization has predominantly been used within software engineering to support the latter phases of the development lifecycle. For example, to depict program call graphs, to visualize source code and to assist with overall program comprehension [3, 25]. Visualizations have also been developed to support testing and debugging activities, and this includes a number of interesting approaches for visualizing bug databases [11, 24]. More recently, visualizations have been developed to show the community contributions underlying the evolution of open source software development projects [32]. Many typical project management activities are also supported by elaborate visual renderings, notably in the form of dashboards that display information about project progress and related performance metrics, using a variety of pie charts, bar charts, graphs and dials [19]. Similar ideas pertaining to requirements measures have found their way into commercial requirements management tools [4].

Given that requirements engineering is that aspect of software engineering that frequently demands intense communication amongst multiple stakeholders in order to uncover and agree upon the needs for a new system development or for a system upgrade, and given the apparent communicative value of a 'good' visual representation, it is surprising that first International Workshop on Requirements Engineering Visualization was only held in 2006. A cursory survey of the requirements engineering literature prior to that date reveals that visualization has mainly been used to support three aspects of requirements engineering practice: (1) to convey the structure of and relations between an evolving set of requirements and other software artifacts, to support the organization of requirements and the management of change; (2) to assist with requirements elicitation sessions and related analysis activities; and (3) to model subsets of the requirements or special properties of these requirements for particular analytical purposes.

(1) Structure and relationships. Commercial requirements management tools have been using representations in symbolic visual forms for many years to depict the hierarchic structure of requirements documents and to make explicit the numerous interrelations between the requirements therein. Such visualizations generally take the form of simple tree structures or more complex connected graphs, the purpose being to assist in the collaborative writing, organization and use of requirements documents [2]. Also, requirements traceability matrices are regularly created to convey linkages between artifacts and support change impact analysis [14].

However, the state of the art does not appear to have advanced much over two decades. For example, the visual conventions incorporated in one of the earlier tools (DOORS), inspired by the classic visualization tome of Tufte[1] [42], are still evident in the tool's interface today. From a management and control perspective, such tools now provide a way to display requirements metrics visually (e.g., the number of changed or implemented requirements), but this is largely by adopting the dashboard approach in which there is often little conceptual correlation between the data being represented and the representations themselves. Whether the various visual mechanisms employed within these tools actually help stakeholders form a deeper understanding of the requirements and processes related to them is debatable. It is clearly not obvious how well these have been designed for use.

(2) Elicitation support. Visual prototypes, storyboards and mock-ups are frequently used in require-

---

[1] Personal communication with one of the DOORS originators in the early 1990's (note that DOORS is now owned by Telelogic) [39].

ments elicitation and analysis sessions to help explore requirements with stakeholders, to clarify understanding and to help reach shared agreement. These visual artifacts can be as crude as hand-drawn sketches on paper napkins through to elaborate interactive experiences provided via dedicated prototyping and scenario presentation tools. One of the earliest systems analysis and design methodologies took a different approach and was centered on the initial creation of a shared visualization called a 'Rich Picture' [8]. This was a freehand sketch intended to describe and understand a complex problem situation prior to undertaking any subsequent analysis. This captured a situation, provoked thinking and remained as a grounding artifact throughout the development process for all stakeholders. Where the term 'requirements visualization' tends to be used interchangeably with tool-supported prototyping [46], this can lead to visual artifacts playing a more transient and throw-away role.

(3) Modeling. A focus of requirements engineering research is to provide a visualization of requirements specified in a formal language, to facilitate validation activities and to increase their general accessibility [40]. Visual modeling also forms a central component of emerging requirements engineering approaches, such as with the strategic dependency and rationale models of the i* framework [51]. Recent research has been examining ways to use visual and spatial cues within these i* models to highlight non-functional quality attributes and to support trade-off analysis [18].

The various diagrams and models of the Unified Modeling Language (UML) clearly provide for visual representations of standard requirements information and regularly feature in requirements documentation. Current requirements engineering research is seeking improved ways to convey visually how such models implement and realize requirements [26]. Although the UML constitutes a major advance in terms of agreed conventions for the form and syntax of particular groups of visual symbols (the chosen set of UML diagrams), there is still a need to integrate their disparate underlying models. This is particularly the case with requirements because of the difficulties associated with the transition between a 'consequent model' of some antecedent subject, such as requirements, and a 'precedent model' of some subsequent object, such as an implemented system [31].

What is common to the existing use of visualization in support of requirements engineering activities is the rare focus on the design of the visualization as a primary artifact, an example being the UML use case diagram, with a clear understanding of the stakeholders and their goals. Equally, little supporting data has been gathered to determine how useful these visualizations actually are. We suggest that the visual artifacts that are evident in this domain are rarely designed explicitly to help stakeholders 'see' requirements and their properties more clearly (in the context of Berger [5]). Without a framework through which to compare and measure the effectiveness of visualizations there is little impetus to seek out new and better approaches. The role of visualization in the requirements engineering field falls a long way behind other areas in which information visualization has been successfully exploited [7].

4. Visualizing Requirements

Despite the supposed popularity of visual modeling languages, requirements still tend to be written in a textual and narrative format [1, 27]. This is the case even when they take the form of user stories, as per the more agile approaches to software development [10]. Requirements documents typically provide a brief natural language description of each requirement, accompanied by data to characterize various attributes about it and so provide for its context. Such meta-data generally includes attributes such as priority, source, test case, cost, rationale, etc. In this way, requirements are rarely stand-alone descriptions; rather, they are multi-dimensional clusters of data. Whether these data are held within a table, spreadsheet or database, the accessibility and use of the fuller contextual information can be challenging, raising the matter of whether their very specification is often only a proforma. The research question is -- how to present these meta-data such that the information that is needed for particular stakeholders and their decision making tasks literally 'pops out' [47] of a designed communicative artifact?

We suggest that a visualization system could take a set of requirements represented in this traditional textual and attribute-rich form, supplemented by the structured UML diagrams often used to augment these descriptions, and render them visible in such a way as to promote shared comprehension of the full set of requirements under study and provoke insight on a number of requirements-related queries (as per recent tools that reveal patterns of change and negotiation within collaboratively authored documents [45]). This relies upon any such visualization being up to date, so an underlying assumption is the existence of through-life traceability (which is outside the scope of this current paper). For instance, a customer may want to assess the relative cost, priority and risk of different requirements at a glance to be aware of quick win areas and to determine where effort is being expended as a project advances. A requirements engineer may want to ensure that all the requirements are grounded in authoritative and representative sources, and establish that all the constituencies have a voice in the requirements, to

form an assessment of coverage and stability. A developer may want to be cognizant of those requirements with many underpinning assumptions or to understand where dynamic changes are occurring to help decide where to focus effort next.

Most requirements engineering resources that offer advice on how to write quality requirements include criteria that are desirable for individual requirements and criteria for sets of requirements as a whole [1, 12, 27, 49]. For example, individual requirements should be: *required* – a stakeholder exists who wants or values the requirement; *correct* – the requirement expresses a valid and desired result; *unambiguous* - the requirement can only be interpreted in one way; *verifiable* - a test exits to check whether the requirement has been met; *understandable* - non-specialists can understand the requirement; and *traceable* – the origins and targets of the requirement are clear. In addition, the entire set of requirements should be: *complete* - all the requirements are specified, with no information missing; *consistent* - no internal conflicts; and *modifiable* - able to be changed. While such criteria are regularly cited as desirable, there is no current way to form an assessment on these qualities in a quick and straightforward observational way. For example, to gain insight on credibility (i.e., reliable sources), feasibility (i.e., some combination of cost, priority, risk and assumptions) and evolvability (i.e., operational traceability). Multi-dimensional clusters of requirements data hold the means to make such assessments, but trawling documents or running database queries to gain requisite information takes time, as well as addresses each aspect in a partial and hence disconnected way. The research question becomes -- can we demonstrate the degree to which requirements have a number of important quality dimensions simultaneously, and in a visual way, using these meta-data? The work that has been done on the visualization of multi-dimensional databases is relevant to answering this question (e.g., [38]).

5. Getting to 'See' Requirements

Software is a complex dynamical system. So is a human. Each is composed of a collection of objects (classes vs. organs) designed to fulfill a specific and typically orthogonal set of requirements that communicate through a well defined network of pathways. When a doctor needs to investigate the health of a patient, an interview goes only so far. A formal diagnosis requires a physical examination in which a patient's anatomical (structural) and physiological (behavioral) features are visually and haptically inspected and monitored. The doctor looks for what and where changes have occurred since the last examination (the

baseline), measuring the patient's height, weight, temperature, pulse, blood pressure, and changes in color, texture and density. Further quantitative and more definitive measures include blood tests, electrocardiograms, radiological imaging (e.g., X-ray and magnetic resonance) and histological analyses (e.g., biopsies). Indeed, what is significant is that many medical tests either involve a direct visual analysis or create images as part of test procedures.

We see the medical metaphor as a possible starting point for constructing analogous visual representations within the requirements visualization process, thus enhancing the visual richness and communication abilities of text and UML. A system's UML descriptions already define its anatomy and physiology and, as such, provide a first approximation of a map of the arrangement and interconnection of classes (organs). Each conceptual class, like an organ, may require different kinds of tests. But, if a query produces data that can be charted (e.g., pie chart of the distribution of development cost estimates), then the image reduces to a cartographic style map with a chart located at each class position. Such maps are the grist for GIS (Geographical Information Systems) and spatial database systems; and the visualization problem reduces to the selection of graphical elements that best display the data components and their relationships (Bertin [6]).

However, this naïve mapping may be insufficient for a number of reasons. First, embedded charts may be difficult to see and quickly interpret within the complex UML forest. Therefore, the visualization system that creates this mapping must support representations that minimize visual complexity to allow perception of important features. As an example, think of a chest X-ray. It is a fuzzy, gray scale image that filters out nearly all structural detail and hence, the structural complexity within the chest cavity. Yet, physicians can quickly diagnose respiratory problems such as lung cancer, tuberculosis and pneumonia just by briefly scanning this image (perhaps having integrated multiple such sources of data). Second, although the data may be either multi-dimensional or quantitative in nature, all that may be required for perception is an image that conveys a key scalar attribute. For example, temperature is an attribute of the dynamics systems such as fluids (e.g., boiling water) or a gauge of the health of an individual (e.g., fever). In this case, the design of the visualization focuses on the monitoring of attributes as opposed to their in-depth study. Such practice is routine in medicine, with scalar and simple vector quantities such as temperature, respiration, and heart monitors recorded and displayed.

Third, attributes may be quantifiable but difficult to convey succinctly. For example, multi-spectral images of the earth or scientific simulations of systems with

thousands of degrees of freedom cannot be adequately represented by directly mapping them to geometry and color. Significant reductions in scale and remapping must be performed before the data may be visualized. Finally, some concepts do not have a direct graphical mapping. Concepts such as love, ambiguity, understandability and correctness must be transformed into a representation that communicates the concept's sense and value. In sum, regardless of the nature and complexity of the concepts that define a system, it may be necessary to transform them into another conceptual form so as to make them more readily perceptible. This transformational process amounts to finding a metaphor that aptly represents and communicates the information to be perceived.

A metaphor is "*something regarded as representative or suggestive of something else, especially as a material emblem of an abstract quality, condition, notion, etc.; a symbol, a token*" [33]. Metaphors make an analogy between the attributes of a known sign or symbol and the comparable attributes of what is to be represented. In short, a metaphor takes what we know and connects it with what we want to know. Metaphors are the foundation for the creation of many models that are fundamental to science and engineering [22]. The power of a good metaphor is that it makes an immediate and instinctive visceral connection with the viewer and triggers an instantaneous response [29]. The advertising industry goes to great lengths to find just the right metaphors [34].

Metaphor selection for visualization may be a challenging process, given the kinds of data to be visualized, the goals of the visualization and the target audience. One approach to selection has been proposed by Eppler and Burkhard in their analysis of knowledge visualization [17]. Inspired by real-world objects and systems, their metaphors fall into four generic groups:

1. *Natural phenomena*: mountain, iceberg, tree, waterfall, volcano, river, cave, etc.
2. *Man-made objects:* balance, ladder, wheel, road, bridge, funnel, umbrella, bucket, lever, etc.
3. *Activities*: climbing, walking, reaching, driving, eating, fishing, harvesting, juggling, pouring, etc.
4. *Abstract concepts*: family, peace, law, chaos, etc.

These metaphors run the gamut from simple (e.g., lever, bridge) to complex (e.g., law, chaos), presenting their own challenges of implementation. Yet, many are already used within the software industry, including tree (data structures), bucket (sorting), waterfall (software engineering lifecycle), iceberg (maintenance) and hill climbing (optimization algorithms).

Whatever the selection, we believe that for a metaphor to be a useful mapping, it should be intuitively recognizable and understandable, simple to implement,

support binary decision making, as well as provide the ability to assess a degree of change in state information. A potential example is the oscillator. Oscillators are objects or systems that demonstrate a systematic variation or fluctuation in a property about a central value. Pendulums and masses on springs are two simple examples. Oscillators may be used to model a wide variety of phenomena including electromagnetic waves, molecular vibrations, population dynamics, climate changes, and planetary motions, to name but a few. None of these systems share the same physics of scale, yet all may be modeled using an understandable metaphor that becomes the underlying conceptual model for defining, quantifying and visualizing these phenomena.

Metaphor selection creates an additional step in the visualization process. It is now composed of three parts that begins with the selection of a suitable metaphor to represent each attribute or concept, followed by the creation of an AVO, and ending with instantiation of the AVO in an appropriate medium. As an example, consider an artist who wants to create an artwork representing his love for his wife. He selects a typical metaphor for love - a red heart. He sketches out a design for an AVO of the heart that defines its size, shape, color and texture. He then paints the heart with oils on canvas. Later his wife, who is a choreographer, decides to craft a dance signifying her love for her husband. Instead of using a heart, she uses cupid as one of many metaphors in her production. Her AVOs specify the kinds of dancers, how they are dressed, their movements and the scene in which they perform. A producer and director then instantiate this work as a performance. What these two examples demonstrate is that there may be more than one useful metaphor for describing a concept, and metaphor selection may depend on the type of AVO constructed. For example, although leaping hearts (yet another metaphor) may make for an interesting performance, dancing cupids may work better in the context of total AVO design. Yet, cupids dancing with hearts plays into the subject of redundancy, an important tool for reinforcing communication; specifically, where multiple channels are used to convey critical information (e.g., flashing lights and sirens). Thus, there is all the more reason to include multiple metaphors when necessary.

Issues of interaction and dynamics among AVOs return us to the map. The UML diagram with which we began Section 5 is static and displays no information about the collaboration among requirements. It may be transformed into a dynamical representation by invoking a molecular model metaphor (like those supported by physical chemistry modeling software [48]). A molecular model represents molecules as a graph in three-dimensions in which vertices represent the location of

atoms and edges represent bonds. Atomic attributes such as size or charge may be mapped onto AVOs as spheres of appropriate size and color. Bonds between atoms may be represented as cylinders in which strong bonds are short and weak bonds are long. Molecules vibrate, their atoms moving in synchrony. These dynamics are usually displayed as animations (oscillations). By analogy, the size of each class's underlying requirements determines it size; the type, its color; and the strength of the coupling between requirements underlying different classes by the distance between them. If classes work in concert to convey and/or to satisfy a requirement, they can be shown to vibrate in a synchronous way; if not, they may be displayed as moving randomly. In both cases, the amplitude of motion may be tied to the rate or degree of change that a class's requirements have undergone over time.

6. Sample Requirements Visualizations

In this section, we outline three approaches to visualization, drawing upon the ideas in the previous sections. Since we are interested in the multi-dimensional nature of requirements (i.e., the use of their meta-data), we base our examples on requirements that have been pre-specified using the Volere requirements shell (requirements that can be found within the Robertsons' text [36]). Here, a textual requirements description is augmented by twelve additional attributes (as described in the text below).

6.1 Symbolic Approach

The Footprint Visualization (Figure 1) is designed to assist the simple assessment of each individual requirement in terms of the presence of all the specified attributes, a form of completeness, and the presence of any apparently inflated attributes, an approximation of complexity. Both factors should indicate the need for further specific examination and, more generally within a set of requirements, its state of development. Using this visualization, every requirement has its own unique shape, a fact that may lead to the recognition of 'healthy' prints and also support diagnostic activities.
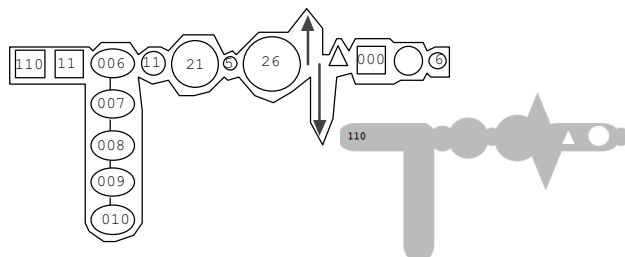

Figure 1. Footprint Visualization.

In the Volere requirements shell, all attributes are initially void and at least five may be vulnerable to inflated entries: list of events/use cases, description, rationale, fit criterion and list of conflicts. Two other attributes, supporting materials and history, may also be vulnerable. In Table 1, attributes 3,4,5,7 and 11 are associated with symbols which may expand in order that they may represent content whose size is not fixed.

Table 1. Volere attributes and representation.

#	Attribute	Type	Content	Symbol
1	requirement number	number	000	square
2	requirement type	number	00	square
3	event/use case list	references	000-000-000-...	linked ovals
4	description	text	abc...	expanding circle
5	rationale	text	abc...	expanding circle
6	source	reference or text	000/abc...	square/expanding circle
7	fit criterion	text	abc...	expanding circle
8	customer satisfaction	range	1,2,3,4,5	upward vertical arrow
9	customer dis-satisfaction	range	1,2,3,4,5	downward vertical arrow
10	priority	range	not specified	upward vertical arrow
11	conflicts list	references	000-000-000-...	linked squares
12	supporting materials	references	000-000-000-...	linked circles
13	history	text, list or references	abc.../000-000-000-...	expanding circle /linked circles

The visualization shows the values associated with each attribute set out as a row of symbols arranged in the simplest form of 'rectilinear network' as classified by Bertin [6]. A single drawn outline encloses all the symbols and creates a more amorphous form which swells and contracts according to symbol sizes and also contains voids where attributes have no value. In the classification of Bertin, this outline would also be a 'network' but one of 'irregular arrangement' without explicit distinction between 'component' and 'relationship'. This outline readily transforms into a single shaded shape punctured where void attributes occur, a form of 'footprint'. Using this approach, a single requirement from page 159 of the Robertsons' text [36] is mapped into the visualization of Figure 1. The symbols in Figure 1 have been generated as follows:

1. Requirement number (110)
2. Requirement type (11)
3. Event/use case list (006)-(007)-(008)-(009)-(010)
4. Description (11 words)
5. Rationale (21 words)
6. Source (5 words)
7. Fit criterion (26 words)
8. Customer satisfaction (3)
9. Customer dissatisfaction (5)
10. Priority (not given)
11. Conflicts list (000)
12. Supporting materials (void)
13. History (6 words)

6.2 Iconic Approach

The Smiley Faces Visualization (Figure 2) is designed to enable a crude assessment on whether certain data are present for requirements, another form of quality checking. A face is generated for each requirement attribute and each requirement is represented by a row of faces in a table. Note that only four attributes are displayed for space reasons. Different attributes are not mapped to distinct features on a single face because facial glyphs designed in such a way can be a slow visualization to read [28]. Only the mouth is used here.

Figure 2. Smiley Faces Visualization.

Two shapes are used to reflect the type of the requirement. A circle is used for functional requirements and a pentagon is used for non functional requirements (since some of these requirements have sharp edges). The attributes we focus on are: *Value* - a combination of customer satisfaction and dissatisfaction. If the values align (i.e., extremely pleased if the requirement is implemented and extremely displeased if not), the mouth smiles. If uninterested, the mouth is straight. If these two values are conflicted in any way, there is a problem to resolve and the mouth frowns (obviously the case with all the requirements above). *Source –* where the source is a named individual within a stakeholder group the mouth smiles; where one of these pieces of data is missing, the mouth is straight; where no data is provided, the mouth frowns. *Rational* and *Fit Criteria –* the presence or absence of these data is signaled by a smile or a frown respectively. By coupling the expression of the mouth with the color of the icon (yellow for happy, grey for ambivalent, blue for unhappy), we provide for redundant coding. Whilst a simple mapping, it is straightforward to read off (quickly) those requirements to pay more attention to.

6.3 Metaphorical Approach

The Volcanic World Visualization (Figure 3) uses a simple metaphor to convey information pertaining to the stability of a set of requirements. It focuses on mapping a subset of the attributes into the visualization to help answer the questions: "Which requirements are likely to blow? Where are the impending storms?" It is designed to illustrate the open-ended possibilities for requirements visualization that are yet to be explored.
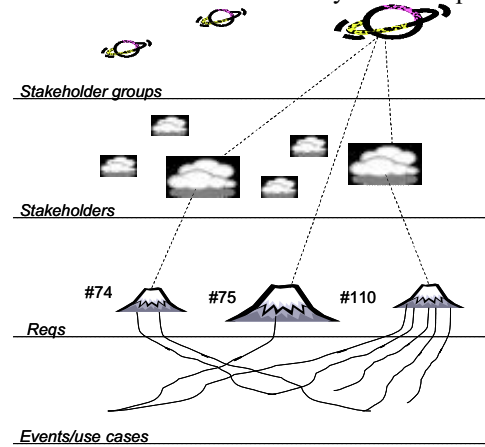

Figure 3. Volcanic World Visualization.

Each requirement is represented as an island with a small volcano. If there are dependencies between requirements, the islands are connected with causeways or clustered, dependent on the nature of the dependency. The type of the requirement is not signified in the visualization but this could be achieved by the shape or color of the volcano. The size of the volcano is proportional to the amount of supporting material, implying there may be effort associated with understanding the requirement (climbing a paper mountain).

A cloud over the volcano signifies a named source (dotted line). The more frequently a source appears in a collection of requirements the larger the cloud becomes. Clouds get larger as air becomes unstable, eventually forming thunder clouds. A thunder cloud would reflect the dominance of an individual source and an impending storm if they were to leave the project. The overarching planetary system reflects the stakeholder groups that the sources belong to. A planet grows in size as a group's voice becomes strongly heard in the requirements, until it overshadows the world. It is straightforward to read off from this visualization whether one stakeholder or group is dominating the requirements and whether another group is not represented at all. These are sources of instability.

The history of a requirement is not represented here, but this attribute is also relevant to the stability of a requirement. Lots of change generally signifies volatility and we propose to represent this by motion. Magma within a volcano erupts based on the movement of tectonic plates. The events or use cases that need the requirement are these tectonic plates and, as they move, rumblings may occur within the volcano.

7. Discussion and Research Agenda

The ideas presented in this paper are preliminary and conceptual, and presented to stimulate discussion about possible directions for research in requirements visualization. Our vision is of a visualization system that automatically maps textually written multi-dimensional requirements (and perhaps data from accompanying UML diagrams) into visual artifacts, permitting stakeholders to actually 'see' the requirements and gain awareness on properties to support high-level decision making. The longer term vision is to provide stakeholders with a way to 'sense' the essential characteristics of these requirements in a more direct and engaging manner to support diagnostic activities.

There are a number of steps that are required to develop this research and realize our near-term vision, notwithstanding the need to be clear about our use of the term 'multi-dimensional'. Multi-dimensional requirements refer to the multiple components (attributes) of content (meaning) associated with requirements or groups of requirements. While lists of 'useful' requirements meta-data abound, there is a need to analyze the most adopted in practice and understand the stakeholder tasks for which these are both required and used. It is possible that many are given default values or merely specified to comply with standards, thus providing little analytical value. It is necessary to understand the insight that use of these values actually (or could potentially) provide prior to designing a suitable visualization process. Choice of meta-data and tasks is thus the first task. Multi-dimensional is also used in the context of multiple visual forms and mapped to the multi-dimensional requirements elements noted above. While some suggestions have been made in this paper, suitable visual forms for each of these elements would need to be investigated rigorously via empirical study.

We plan to build a prototype based upon the outcomes of the above research steps and to compare the designed visual artifacts with requirements represented in the more traditional form. We will evaluate the impact of a shared communicative artifact that provides for situational awareness on a small set of representative stakeholder tasks (to be identified). We anticipate that issues of dimensionality and scale (i.e., number of requirements to feature in the visualization) can be addressed through filtering mechanisms and considered selection of preattentive cues [9, 21], to focus the visualization according to task and salient attributes. We envisage that this research will lead to visual cues or patterns that can help to ascertain the ongoing health of a project's requirements and so inform better practice.

This paper has outlined a vision for using visualization techniques in the field of requirements engineering. It has outlined the process of constructing a visualization system and suggested an approach based on metaphor and mapping. It has also provided some simple examples that show the many directions such work could take. Merely looking is a passive form of engagement with requirements. We claim that there is a need for research that will lead to stakeholders seeing requirements and, ultimately, to experiencing requirements [23, 37, 43, 50], especially if they are to make quick and informed judgments about them.

8. References

[1] Alexander, I.F. and Stevens, R. *Writing Better Requirements*, Addison-Wesley, 2002.
[2] Austin, M., Mayank, V. and Shmunis, N. PaladinRM: Graph-based Visualization of Requirements Organized for Team-based Design, *Systems Engineering Journal*, Vol. 9, Issue 2, May 2006, pp.129-145.
[3] Ball, T. and Eick, S.G. Software Visualization in the Large, *IEEE Computer*, Vol. 29, Issue 4, April 1996, pp.33–43.
[4] Baxter, P. and Tavassoli, D. *Management Dashboards and Requirements Measurement*, Telelogic White Paper, Version 1, 1 June 2006.
[5] Berger, J. *Ways of Seeing*, British Broadcasting Company and Penguin Books, 1972.
[6] Bertin, J. *Semiology of Graphics*; translated by W.J. Berg. Madison, Wisconsin: University of Wisconsin Press, 1983.
[7] Card, S.K., Mackinlay, J.D. and Shneiderman, B. *Readings in Information Visualization: Using Vision to Think*, Morgan Kaufmann Publishers, Inc., 1999.
[8] Checkland, P.B. *Systems Thinking, Systems Practice*, John Wiley and Sons Ltd., 1981.
[9] Chernoff, H. Using Faces to Represent Points in K-Dimensional Space Graphically, *Journal of the American Statistical Association*, 68 (342), 1973, pp. 361–368.
[10] Cohn, M. *User Stories Applied: For Agile Software Development*, Addison-Wesley Professional, 2004.
[11] D'Ambros, M., Lanza, M. and Pinzger, M. A Bug's Life: Visualizing a Bug Database, *Proc. 4th International Workshop on Visualizing Software for Understanding and Analysis (VISSOFT 2007),* Banff, Canada, 24-25 June 2007, pp.113-120.
[12] Davis, A.M. *Software Requirements: Analysis and Specification*, Prentice-Hall, Inc., 1990.
[13] Delgadillo, L. and Gotel, O. Story-Wall: A Concept for Lightweight Requirements Management, *Proc. 15th IEEE International Requirements Engineering Conference (RE 2007)*, Delhi, India, 15-19 October 2007 (to appear).
[14] Duan, C. and Cleland-Huang, J. Visualization and Analysis in Automated Trace Retrieval, *Proc. 1st International Workshop on Requirements Engineering Visualization (REV 2006),* Minneapolis, Minnesota, USA, 11 Sept. 2006.
[15] Earnshaw, R.A. *An Introductory Guide to Scientific Visualization*, Springer-Verlag, 1992.
[16] Eppler, M. Making Knowledge Visible Through Knowledge Maps, in C.W. Holsapple (Ed.), *Handbook on Knowledge Management*, Berlin: Springer-Verlag, 2002, pp.189-206.

[17] Eppler, M. and Burkhard, R. Knowledge Visualization, 2004 (available from *http://www.knowledgemedia.org /modules/pub/view.php/knowledgemedia-6*7).

[18] Ernst, N.A., Yu, Y. and Mylopoulos, J. Visualizing Non-Functional Requirements, *Proc. 1st International Workshop on Requirements Engineering Visualization (REV 2006),* Minneapolis, Minnesota, USA, 11 September 2006.

[19] Few, S. *Information Dashboard Design: The Effective Visual Communication of Data*, O'Reilly, 2006.

[20] Haber, R.B. and McNabb, D.A. Visualization Idioms: A Conceptual Model for Scientific Visualization Systems, in *Visualization in Scientific Computing*, G.M. Nielson, B. Shriver and L.J. Rosenblum (Eds.), IEEE Computer Society Press, 1990.

[21] Healy, C.G., Booth, K.S. and Enns, J.T. High-Speed Visual Estimation Using Preattentive Processing, *ACM Transactions on Human-Computer Interaction,* Vol. 3, No. 2, 1996, pp.107-135.

[22] Hesse, M.B. *Models and Analogues in Science*, Notre Dame, IN: University of Notre Dame Press, 1966.

[23] Hornecker, E. and Buur, J. Getting a Grip on Tangible Interaction: A Framework on Physical Space and Social Interaction, *Proc. SIGCHI Conference on Human Factors in Computing Systems*, Montréal, Québec, Canada, 22–27 April 2006. R. Grinter, T. Rodden, P. Aoki, E. Cutrell, R. Jeffries and G. Olson, (Eds.), CHI 2006, ACM Press, New York, NY, USA, pp.437-446.

[24] Jones, J.A., Harrold, M.J. and Stasko, J. Visualization of Test Information to Assist Fault Localization, *Proc. 24th International Conference on Software Engineering (ICSE 2002),* Orlando, Florida, USA, May 2002, pp.467-477.

[25] Knight, C. and Munro, M. Virtual but Visible Software, *Proc. 4th IEEE International Conference on Information Visualization,* London, UK, 19-21 July 2000, pp.198–205.

[26] Konrad, S., Goldsby, H., Lopez, K. and Cheng, B.H.C. Visualizing Requirements in UML Models*, Proc. 1st International Workshop on Requirements Engineering Visualization (REV 2006),* Minneapolis, Minnesota, USA, 11 Sept. 2006.

[27] Kovitz, B.L. *Practical Software Requirements: A Manual of Content and Style*, Manning Publications Co., 1998.

[28] Lee, M.D., Reilly, R.E. and Butavicius, M.A. An Empirical Evaluation of Chernoff Faces, Star Glyphs, and Spatial Visualizations for Binary Data, *Proc. Australasian Symposium on Information Visualisation*, Adelaide, 2003.

[29] Marchese, F.T. OpGlyph: A Tool for Exploring Op Art Representation of Height Field and Vector Field Data, *Proc. Working Conference on Advanced Visual Interfaces (AVI 2002),* Trento, Italy, May 2002, ACM, pp.103-107.

[30] Morgan, R. and Maurer, F. MasePlanner: A Card-Based Distributed Planning Tool for Agile Teams, *Proc. IEEE International Conference on Global Software Engineering (ICGSE 2006),* Florianopolis, Brazil, 16-19 October 2006, pp.132-138.

[31] Morris, S.J. and Gotel, O.C.Z. Model or Mould? A Challenge for Better Traceability, *Proc. International Workshop on Modeling in Software Engineering (MiSE 2007)*, Minneapolis, Minnesota, USA. ACM, May 2007.

[32] Ogawa, M., Ma, K-L., Bird, C., Devanbu, P. and Gourley, A. Visualizing Social Interaction in Open Source Software Projects, *Proc. 6th International Asia-Pacific Symposium on Visualization (APVIS 2007)*, Sydney, Australia, 5-7 February, 2007, pp.25-32.

[33] Oxford English Dictionary, 1989.

[34] Reichert, T. and Lambiase, J. (Eds.). *Sex in Advertising: Perspectives on the Erotic Appeal*, Lawrence Erlbaum, 2002.

[35] Robertson, P.K. A Methodology for Choosing Data Representations, *IEEE Computer Graphics and Applications*, Vol. 11, No. 3, May 1991, pp.56-68.

[36] Robertson, S. and Roberson, J. *Mastering the Requirements Process*, ACM Press, 1999, p.9, p.157, p.159 (also, *http://www.systemsguild.com/GuildSite/Robs/Template.html*).

[37] Salisbury, K., Conti, F. and Barbagli, F. Haptic Rendering: Introductory Concepts, *IEEE Computer Graphics Application*s, Vol. 24, No. 2, 2004, pp.24-32.

[38] Stolte, C., Tang, D. and Hanrahan, P. Polaris: A System for Query, Analysis, and Visualization of Multidimensional Relational Databases, *IEEE Transactions on Visualization and Computer Graphics*, Vol. 8, No. 1, January-March 2002, pp.52-65.

[39] Telelogic. DOORS, *http://www.telelogic.com/corp/ products/doors/index.cfm* (accessed August 2007).

[40] Teyseyre, A. A 3D Visualization Approach to Validate Requirements, *Proc. Congreso Argentino de Ciencias de la Computación*, Argentina, October 2002.

[41] Tory, M. and Moller, T. Rethinking Visualization: A High-Level Taxonomy, *Proc. IEEE Symposium on Information Visualization (INFOVIS 2004)*, IEEE Computer Society, Washington DC, USA, 10-12 October 2004, pp.151-158.

[42] Tufte, E.R. *Envisioning Information*, Graphics Press, 1990.

[43] Ullmer, B. and Ishii, H. Emerging Frameworks for Tangible User Interfaces, *IBM Systems Journal*, Vol. 39, No. 3 and 4, 2000, pp.915-931.

[44] United States Joint Forces Command Glossary, *http://www.jfcom.mil/about/glossary.htm* (accessed August 2007).

[45] Viégas, F.B., Wattenberg, M. and Dave, K. Studying Cooperation and Conflict between Authors with *history flow* Visualizations, *Proc. SIGCHI 2004*, ACM Press, Vienna, Austria, 2004, pp575-582.

[46] Wadhwa, S. Visualizing Requirements, *Requirements Quarterly: The Newsletter of the Requirements Engineering Specialist Group of the British Computer Society,* RQ42, December 2006.

[47] Ware, C. *Information Visualization: Perception for Design*, Morgan Kaufmann Publishers, 2004.

[48] Wavefunction, Inc. Spartan Software. *http://www.wave-fun.com/products/spartan.html* (accessed September 2007).

[49] Wiegers, K.E. Writing Quality Requirements, *Software Development*, Vol. 7, No. 5, May 1999.

[50] Wisneski, G., Ishii, H., Dahley, A., Gorbet, M., Brave, S., Ullmer, B. and Yarin, P. Ambient Display: Turning Architectural Space into an Interface between People and Digital Information, *Proc. 1st International Workshop on Cooperative Buildings (CoBuild 1998)*, Darmstadt, Germany, 25-26 February 1998.

[51] Yu, E. Towards Modelling and Reasoning Support for Early-Phase Requirements Engineering, *Proc. 3rd IEEE International Symposium on Requirements Engineering (RE 1997),* Annapolis, Maryland, USA, 6-8 January 1997, pp.226-235.