Dissection of a Visualization On-Demand Server

Romain Vuillemot, Béatrice Rumpler, and Jean-Marie Pinon

Université de Lyon, LIRIS, F-69621, Lyon, France
romain.vuillemot@insa-lyon.fr
http://liris.cnrs.fr/romain.vuillemot

Abstract. In this paper, we detail specifications of a Visualization On-Demand (VizOD) server. We show that packaging information visualization processes into services reachable over a network benefits both users and programmers, by reducing development cycles. We implemented a prototype based on our architecture, resulting in an innovative way to visually explore large movie database. We discuss early results and our main perspective is to federate a community of users and practitioners to better design interactive environments and understand users behaviors.

Keywords: Information Visualization, Service Oriented Architecture, Visualization On-Demand.

1 Introduction

Availability of information visualization techniques are pare always strongly tied to a specific task or application. Making those broadly available to users according to data, needs and task to achieve (see Figure 1) would make more applications available. Availability is meant in terms of time-to-product, skills required and span of choice to offer users the right visualization technique.

Painted with broad strokes, one major *technical* issue with visualization techniques is the data format heterogeneity and the lack of reliability: there is a huge gap between

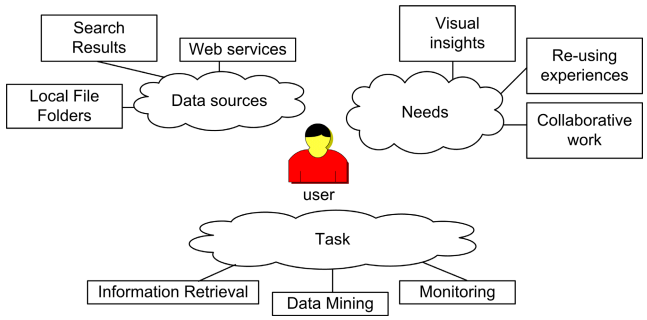

Fig. 1. Users ecosystem is complex with data requirements, various needs and tasks to achieve. Trends show that it tends to be heterogeneous and increasing in size. Our focus is on visualization and interactions, and our approach is to make them available as services.

J. Filipe and J. Cordeiro (Eds.): ICEIS 2008, LNBIP 19, pp. 348–360, 2009.

a proof of concept issued from research works and a out of the shelve tool. Advanced contributions exist, but scattered in so many different application fields. For instance, visual data mining tools are very prolific in biology [1] with stunning results facing real life problems, especially dealing with data masses. New patterns finding heuristics in huge structures are available for a specific tasks, but those inspiring data depiction remain domain specific. At the end of the day, end-users cannot benefit from most of the scientific tools even if they look inspiring and potentially useful. And finally, concerning interactive environments, the desktop metaphor is still and exclusively massively used because of universal availability: innovation cannot make its breakthrough.

As companies massively digitalize data for productivity, ubiquity and quality management, users need to keep track and get analytics on data evolution, issued from multiple sources. Whereas there exists tools to perform dedicated data analysis, as far as we know there is generic visual overview technique to get visual insights, regardless data or tasks (see Figure 2). Our goal is to quickly (in a product development perspective) build or setup an interface to uncover phenomena unseen at the first sight, which will guide users to a specific data subset or projection.

Existing solutions are either restricted to an application or limited in extension. Starting a whole product cycle induces cost, time waste and risk. This process may also decrease users attention and productivity by changing their habits and reflex. New approaches have to keep users in the same interactive environment, with the same interactive devices he masters, just data being different.

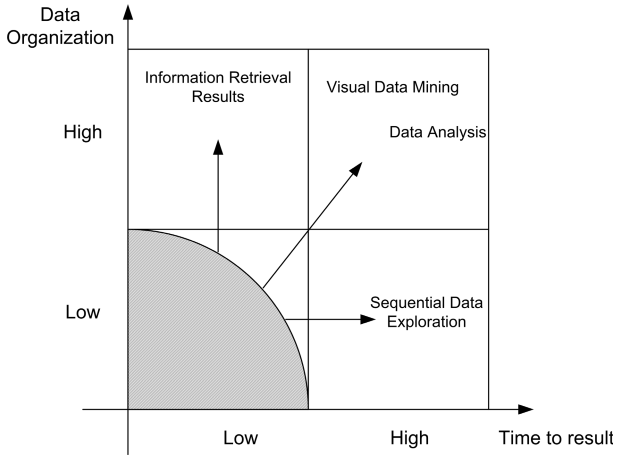

Fig. 2. Users tasks have a broad span of data organization and time to result. Generic overview (gray area) of data structures leads to more specific analysis. We focus on that area which are the root of analytical and discovery processes.

Actors needs are identified as follow:

– **End-Users.** They need abstract data handling to get visual insights of the data, with appealing visual metaphors. Such a process must take into account situations not requiring any symbol being entered into the system (in case users cannot formalize

their needs). Users have different background richness and cultures, so individual characteristics have to be stored.

- **Designers.** They are experts in the field of translating users needs into software or assembly of software coupled with interactive devices. Whereas design knowledge is stored in a guideline format [2], it lacks formalization and evaluation. Re-use of existing libraries and environments becomes crucial with increasing system complexity and heterogeneity. Products life cycle are then extended.
- **Managers.** They are using advanced monitoring tools with high reliability to detect trends. Trends are a way to anticipate the future and can be raised by means of complex and long term analysis..

This paper is organized as follow. Section 2 focuses on similar works in Information Visualization and Service Oriented Architecture. Section 3 focuses on the architecture outlines. Section 4 describes a prototype that has been developed. Section 5 discusses results and perspective. Section 6 concludes.

2 Related Work

Our approach consists in bridging Informations Visualization (InfoVis) techniques and Service Oriented Architecture (SOA). We review related work and a synthetic comparison of the two fields is given (Table 1).

2.1 Information Visualization

Conceptually, the fundamental goal of InfoVis techniques are to find the right visualization at the right moment. A complete visualization process results in outputs such as maps to discover relationships among data. Relationships can be either internal or external, helping users to better understand complex static or dynamic dataset changing over time. Human capabilities are thus enhanced, but limited cognitive memory must be taken into account in order not to overload users [3]. Also, codes or users knowledge is to be integrated to reduce information dimensions. Limited display space must also be considered, and can be done by coordinating multiple views [4] .

Technically, the underlying issue is that visualization is hard coded to data and task to achieve. Today, reusing a technique means starting another configuration/implementation cycle according to informal design guidelines. These recommendations lack of formalization and are given in pattern format which has to be understood by experts, inducing extra costs. Another limit is that visualizations are local to users application and limited to his computing resources. And because contributions are scattered in so many different application fields, there are neither central repository, nor evaluation. Some lists exist, such as Many Eyes[1] or Visual Complexity[2]. The latter consists in an updated screenshot and video repository: but there are no semantic taxonomy providing automatic access to visualizations.

[1] http://services.alphaworks.ibm.com/manyeyes/
[2] http://www.visualcomplexity.com/

Data Transformation Models. Our goal is to understand the visualization process in general, regardless data types, task or application domain. Then a model oriented analysis of the visualization has to be performed [5]. This approach has been commonly used, resulting in many models. We focus on two major models from two distinct fields that are *Information* Visualization [6] and *Scientific* Visualization [7]. See Figure 3.

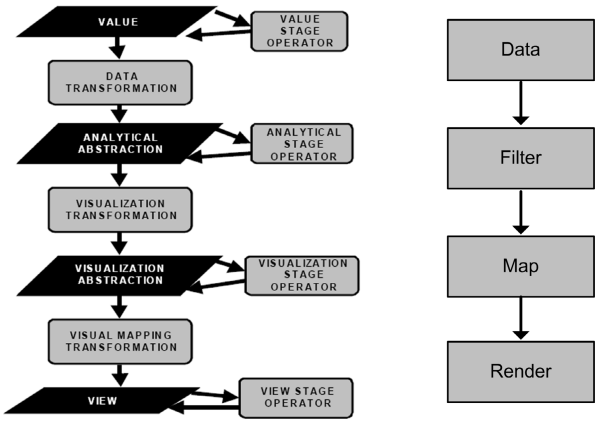

Fig. 3. Existing data transformation models are common in Information Visualization [6] and Scientific Visualization [7]. They help better understanding the various states of data.

We proposed a slightly different model based on the previous ones, including interactions and considering the visualization process as a sequence of independent operations. Our model decomposes the Information Visualization process into 3 stages, each coupled with users actions (see Figure 4). Interactions can be either manual (users action), either semi-manual (users action triggers a system reaction) or automatic (system action).

The model description is as follow:

Extraction: is a step transforming data into an internal such as semantic (RDF), physical (matrix, directories) or even social, which extract different points of view from the data.
Selection reduces datasets by selection (SQL, SPARQL, ..), aggregating and projecting in an appropriate manner, relieving users from a potential overload.

Layout: gives a spatial attribute to abstract data structure such as graph, lists or multi-dimensional sets. The layout can be 2D maps or 3D model.
Organization is an any action changing the layout attribute of data, but not the data themselves.

Render: transforms abstract data layout into images or 3D scenes.
Filtering means to *post*-process (using image analysis techniques such as Gaussian blur or Laplacian filter) to provide a *pre*-processed visual result [8]. These mathematical computations based on 2D-signal transformations help, for instance, users to fade details or highlight contours.

The above mentioned layers descriptions help to identify and describe each technique, that can now be seen as independent modular programs.

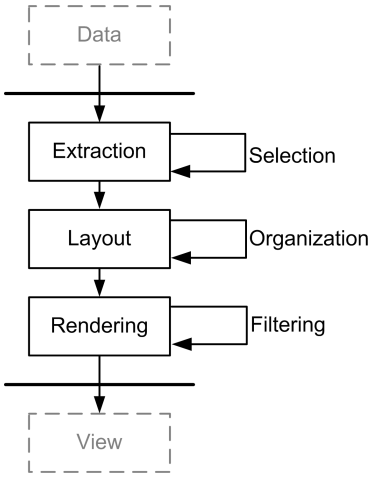

Fig. 4. We introduce a holistic data transformation model, including both data transformation processes and interactions

2.2 Service Oriented Architectures

Service Oriented Architecture (SOA) is an architecture style that aims at reorganizing business processes into loosely coupled packages. The packages are distributed into modules reachable over a network. The simplicity of use and universality of access makes it a design style helping the reuse of self-describing components. The components are interfaced as individual entities with the propensity to build applications very quickly. Services goal is to allow functionality to be stuck together to form ad-hoc applications reusing existing software service. The capabilities of adaptability and evolution are high by adding a new service, reducing developing costs and a quick deployment.

Service are software reachable over a network independently from the underlying implementation. Languages describing them are:

- Interfaces are published in the WSDL file (XML-Based Document that describes how to communicate with the Web Service) [9].
- Service repositories store WSDL files and are using UDDI (Universal Description, Discovery and Integration) helping to match with users needs. [10]
- Clients having retrieved relevant interface will contact the service provider using SOAP [11].

Service composition [12] enables resources to be merged and responds more quickly and cost-effectively to changing market conditions. Service helps not having license/ software distribution issues, and can reduce distribution costs, piracy and reverse engineering.

2.3 Visualization Service

The challenge of generically benefiting from visualization techniques has already been faced through many researches. A prolific field is *Scientific* Visualization, providing

related architectures. [13] introduced a client/server communication based on a simple reference model to perform data visualization. The visualization is done over the web, and focuses only on a specific data type which are plots. [14] introduces a modular visualization environments enabling users to change the data pipeline. A GUI interface is described, where dynamic change of the visualization pipeline can be done by users. Finally [15] is a Scientific Visualization system offering web services facilities, but which is not based on a model. Thus visualization is seen as a whole process which steps can't be isolated.

Table 1. Comparative characteristics of both InfoVis and SOA. Infovis is locally tied to clients, whereas SOA are distributed and broadly available.

Type/Field	**InfoVis**	**SOA**
Location	Local	Distributed
Coupling	Tight	Loose
Flexibility	Bundle	Package
Messages	File format	Messages
Communication	Function calls	Protocol

3 A Visualization On-Demand Architecture

Our contribution is a Visualization On-Demand (VizOD) architecture is 1) to separate into independent modules the visualization process according to our model (as seen on Figure 4) and 2) distribute processes, regardless the data being studied. This modular approach has to be combined according to a strategy, taking into account both technical constraints and users' preferences.

Using *on-demand* paradigm means we consider the rendered data stream as a media, such as movies with Video On-Demand (VoD) that can be chosen according to a user action. Conceptually, this expression is well-fitted since we consider the visualization as a stream of existing knowledge, just with another perspective. Technically, it holds many identical properties as VoD, such as cache, replication and performance.

3.1 Architecture

We now focus on interfaces and communication protocols of each steps we introduced in our holistic model (4).

Processes Subdivision. Making smaller parts (called business processes) of a larger system, operating independently. Each business process has properties (described in a UDDI repository) such as a description including the task it achieves, performance and complexity. These information will be useful to respect users Quality of Service constraints.

While communication among the processes becomes asynchronous, modules keep interdependence constraints. For instance a change in the layout will trigger a new render. A color change in the graph depiction will leave an identical layout, but here again

render will have to be regenerated. We used as exchange protocol among the modules existing intermediate data transformation. For instance, the extraction module will communicate a graph structure to the layout process as it would be done in a monolithic architecture. In other words, external interface mirror internal temporary data residing in computer memory. The language choice turned out to be XML-like to wrap messages as showed on Figure 5. Then an additional SOAP communication protocol layer is added.

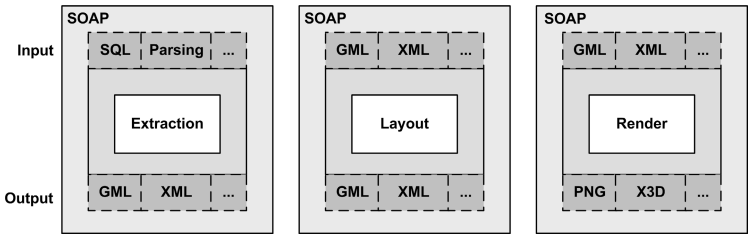

Fig. 5. Modules are independent steps issued from data transformation models. They are reusable and can be remotely located. Modules are encapsulated and communicate using SOAP messages.

Processes Distribution. Business processes can be located at any places, and reachable through a service repository. Process distribution means that some processes may remain local to users (e.g. because of privacy issues) while other may be distributed and reachable other a network (e.g. because they require lots of computing resources).

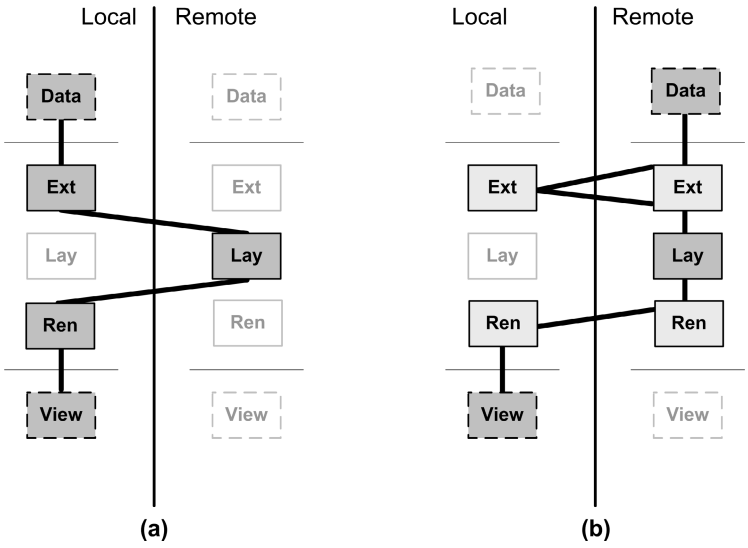

Fig. 6. New strategies appear based on our approach of cutting data transformation steps and providing them as independent services. For instance (a) strategy is using remote layout process, (b) is to keep only local interactions.

There exists many distribution strategies that can be complex and dynamic (changing over time, service availability or service load). Two examples are described on Figure 6: (a) shows that user hosts the dataset and transfer only data structure to perform a layout process on it. This is a case where the layout needs lots of computing power (b) shows an opposite strategy, where users select only the data and interact with the render only. This is a case where a dataset is shared and reached through a local interactive environment.

3.2 Strategy of Access

A strategy is a common way of combining small steps to tackle a bigger problem. A step will be regarded as a group of processes, which are selected and assembled together to solve a task. Assembly of steps will be done as close as the way the mind is working and will be called *patterns*. These patterns are following common orchestration that have been subject of study, such as Schneiderman's Overview Zoom and Details-On-Demand (OZD) [3].

3.3 Personalization

While a company is an organization having the same focus, individuals have specificities to be considered. Even if every task or context may be different, there exists a visual knowledge about data structures (e.g. tree-like structures similar as Figure 8) that are common to every kind of information. Learning how to understand and master this knowledge requires time, but once done it can result in time savings by cutting delays to visually master new datasets. Thus, users visual habits and preferences have to be identified and stored.

Personalization [16] is a way of taking users preferences into account. Researches have been carried out in such direction as in information retrieval systems, in order to reduce datasets according to users explicit or implicit preferences. *Explicit* preferences are users selections and configurations operations, such as local environment choice or service choice. *Implicit* preferences are users history or any typical behavior registered in a non-intrusive way. For instance, if a specific service or group of services are invoked many times, they will be considered as a preference, even if no question has ever been asked to users [17].

Preferences can also vary from short to long term interest. *Short term* preferences are edge colors, any encoded knowledge (such as symbols) and filters of the render which aims at solving a task in a specific context. *Middle term* preference is the data layout which can't vary (whereas colors can) in order not to disturb users mental model, which is his internal vision of a virtual scene. Finally, a *long term* interest concerns the interactive environment in which are integrated visualizations.

Preferences will be stored in a *User Visual Profile* and will be available regardless users location or context.

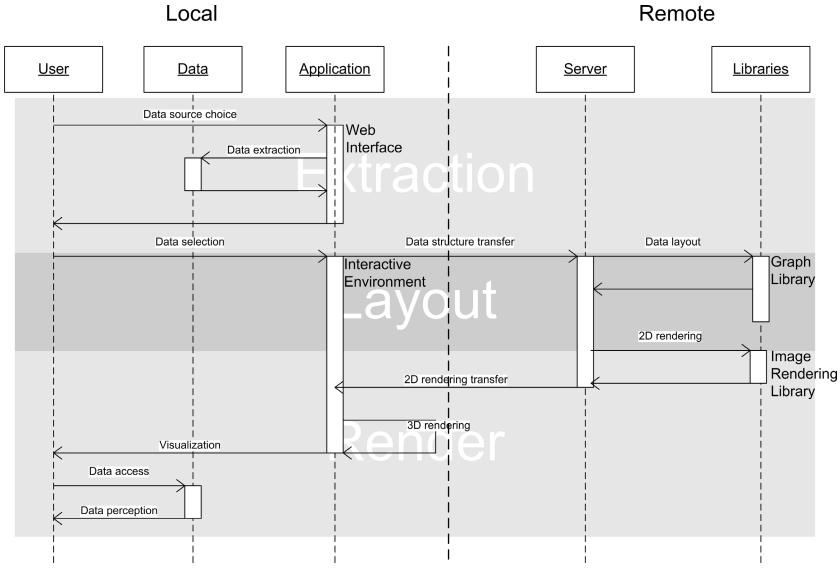

Fig. 7. Sequence Diagram of an application based on the VizOD architecture. Interactions and rendering steps remain local, whereas all other steps are outsourced remotely.

4 Prototype

To validate our architecture, we developed a first VizOD prototype, based on our recommendations and using existing visualization techniques and interactive environments. Advanced technical details are available in [8]. Our approach was to implement the strategy described in Figure 6 (b) which sequence diagram is available (Figure 7).

The dataset is a movie database based on a sample of the Internet Movie Database[3]. The data extraction consists in performing queries, selecting 1) users and 2) for each users their rated movies. That selection is done by means of a web interface allowing SQL-like queries. The extracted result consists in a tree-like structure where the root is an artificial node connecting all users with their rated movies as leaves. The graph layout techniques used was originally aimed to display protein networks [1]. Other graph visualization tools exist such as [18]. The render step results in an image with annotated data (containing details about movies), provided in a separate file structured in a `XML`-like format, `KML` (Keyhole Markup Language). The result is the picture displayed on Figure 8.

The image looks intriguing, but holds only structural information: it has to be included in users interactive environments that makes metadata available (such as nodes and edges names visible).

We selected Google earth (GE) as an interactive environment (with all geo-spatial features disabled). A screenshot is available (Figure 9). GE is installed and run by the client, and connects to VizOD by mean of HTTP requests. Using HTTP helps to keep away firewall issues or any complicated network configuration. VizOD will seamlessly

[3] http://www.imdb.com/

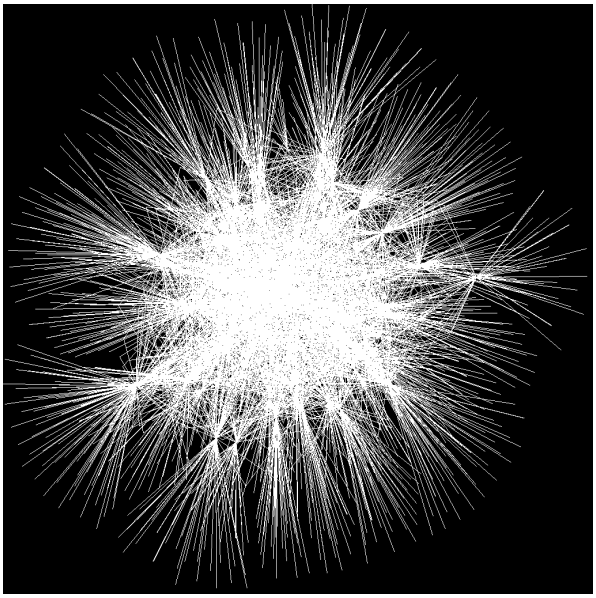

Fig. 8. The image shows a tree-like layout visualization of a query result from a movie database. Images need to be coupled to local interactive environment to let users get more informations such as metadata and annotations.

map onto GE's 3D sphere an image according to users altitude and angle of view, following [3]'s recommendations. The result is a 3-layered multi-scale strategy combining external business processes, and resulting (by decreasing altitude) in an overview layer, zoom layer and details layer. The overview layer aims at showing global trends, then it will be connected to a blurring render service, to remove details and raise trends. The service interfaces Gimp[4] used in command line. The zoom layer remains the original image. The detail layer is the original image augmented with details on top of it (included in the `KML` file). Bandwidth usage has been minimized with that detail layer, by keeping the zoom image locally and only requesting and adding light `KML` data on top of it. The `KML` is converted to additional vectorial graphics (lines, captions, ..) by GE.

We used a server running on GNU/Linux Fedora Core 6, 512 Mo RAM, AMD AthlonTMXP 2500+ (1.8 GHz) 512 KB cache memory. It took 59 seconds to generate a single image. The layout process is the greediest one. Comparatively, generating or blurring images involves nearly no extra time or resource use cost.

Results, issued from experiments, is that GE is a good metaphor since it implements a well-known object that is earth. And users already used GE for other purpose: then we managed to minimize the environment mastering phase by reusing an existing and widespread tool. Usability was excellent since we re-used a powerful environment, which remains very reactive to every gestures and moves from users, even if images were not fully loaded or updated yet. Thanks to our VizOD approach, many

[4] GNU Image Manipulation Program available at: http://www.gimp.org/

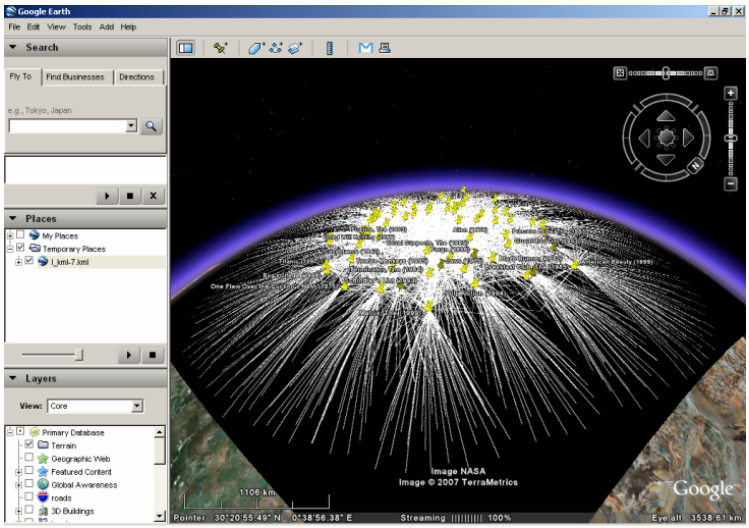

Fig. 9. Mapping rendered images mapped onto a 3D-sphere is an efficient way to provide ends users interactive abilities. Panning, zooming and rotating, coupled with details on demand are powerful way to face visual overloads.

visualization strategies can be adopted, while the client stays focused to a very same interface. The dataset can even totally change without absolutely no interface change. Adding new features on VizOD will be transparent for users. Finally users appreciated to use an attractive means that is a 3D sphere (similar to the *iPod effect* where the appealing wheel attracts users).

5 Discussions and Perspectives

In this section we discuss results applications of our architecture and provide research tracks that need top be investigated.

Company Benefits issued from using a VizOD architecture will come by outsourcing visualization to experts, providing software as services with better support and piracy prevention. The information system rationalization will go further by centralizing computing power at the same remote place and leaving end-users with lightweight heterogeneous terminals. The software maintenance routine is not on site any more but on the VizOD servers, holding business processes, which can be numerous allowing diversity and backups alternatives. New economical models will appear for producers.

Privacy and Security Issues are a big concern in our approach. Regarding the steps of our model one can see that rendered images prevent reverse engineering process. Furthermore, extracting data structures only will give structural information and preventing details being visible: quantity of information is given, not quality. Finally, a service approach prevents implementation details to be visible. However, we keep SOA related issues, such as messages exchange that is prone to attacks.

Users needs have to be Considered Globally, with imperfections, such as short attention spans. Memories are also important aspects to deal with, especially with data masses and in the case information is available in streams which are not stored: users full attention is then required. The focus can be on visualizing updates, data growth, rather than content itself: the change or the behavior becomes as important as the instant content. The scalability and adaptability of the VizOD architecture is crucial.

Services Mashup interfaces are new way for users to compose services to build up and share new ones. But is does not include yet extra process such as data layout and render. A future work is to implement a *Visualization Mashup Interface* to cope with the lack of semantic and integration visualization service repository. Users needs tools in new era where web-users have become actors using web interfaces.

New Design Process and product life cycle will emerge. Programmers are not constrained, then they will keep their own programming habits. A piecewise conception process can be set up, with progressive features. Other Research communities are addressed such as cognitivists in order to observe users behavior, interface designer to re-think the way to design and evaluate. New actors such as artists can now fully take part to the design process by including artistic among one or many navigation step.

6 Conclusions

In this paper we introduced and implemented a visualization on-demand server (VizOD). We first proposed a holistic data transformation model, considering both visualizations and interactions. Every step of the model are considered as business processes that can either remain local or be distributed. Such an approach allows end-users to benefit and personalize visualization services, and couple it into their local interactive environments. A present day result is a prototype resulting from an assembly of existing tools and showing that even non visualization-dedicated tools (such as graph manipulation libraries) can quickly result in an innovating application, following VizOD's specifications, in a context of affordable systems and non-expensive softwares.

Encapsulated processes to make them automatically discoverable and usable is our next research step. There is also a semantic gap between users task needs which are expressed in natural language and tasks the machine already knows how to achieve. To tackle this issue, our next step is building on line communities that will fertilize best practices or novel uses. We will also focus on users generated data (e.g. traces of use) that have to be structured, filtered and sorted. More generally, a stable on line framework has to emerge and be sustainable over time, and then lessons will be learn by widespread use.

References

1. Adai, A.T., Date, S.V., Wieland, S., Marcotte, E.M.: Lgl: creating a map of protein function with an algorithm for visualizing very large biological networks. J. Mol. Biol. 340, 179–190 (2004)
2. Shneiderman, B.: Designing the User Interface: Strategies for Effective Human-Computer Interaction. Addison-Wesley Longman Publishing Co., Inc., Boston (1997)

3. Shneiderman, B.: The eyes have it: A task by data type taxonomy for information visualizations. In: VL 1996: Proceedings of the 1996 IEEE Symposium on Visual Languages, Washington, DC, USA, p. 336. IEEE Computer Society, Los Alamitos (1996)
4. Michelle, W.A., Kuchinsky, A.: Guidelines for using multiple views in information visualization. In: Advanced Visual Interfaces, pp. 110–119 (2000)
5. Butler, D.M., Almond, J.C., Bergeron, R.D., Brodlie, K.W., Haber, R.B.: Visualization reference models. In: VIS 1993: Proceedings of the 4th conference on Visualization 1993, pp. 337–342 (1993)
6. Chi, E.H.: A taxonomy of visualization techniques using the data state reference model. In: INFOVIS, pp. 69–76 (2000)
7. Haber, R., McNabb, D.: Visualization idioms: A conceptual model for scientific visualization systems. In: Nielson, G.M., Shriver, B., Rosenblum, L.J. (eds.) Visualization in Scientific Computing, pp. 74–93. IEEE Computer Society Press, Los Alamitos (1990)
8. Vuillemot, R., Peralta, V.: From Beautiful to Useful: A Multi-Scale Visualization of Users Movie Ratings. Technical Report RR-LIRIS-2008-001, LIRIS UMR 5205 CNRS/INSA de Lyon/Université Claude Bernard Lyon 1/Université Lumière Lyon 2/Ecole Centrale de Lyon (2008)
9. Christensen, E., Curbera, F., Meredith, G., Weerawarana, S.: Web Services Description Language (WSDL) 1.1. Technical report, W3C Note (2001), `http://www.w3.org/TR/wsdl`
10. UDDI: Universal description, discovery and integration, version 3. OASIS, Billerica, Mass. (2000), `http://www.uddi.org`
11. SOAP: Simple object access protocol (soap 1.1) (2000), `http://www.w3.org/TR/SOAP`
12. Agarwal, V., Dasgupta, K., Karnik, N., Kumar, A., Kundu, A., Mittal, S., Srivastava, B.: A service creation environment based on end to end composition of web services. In: WWW 2005: Proceedings of the 14th international conference on World Wide Web, pp. 128–137. ACM, New York (2005)
13. Wood, J., Brodlie, K., Wright, H.: Visualization over the world wide web and its application to environmental data. In: VIS 1996: Proceedings of the 7th conference on Visualization 1996, p. 81. IEEE Computer Society Press, Los Alamitos (1996)
14. Bonneau, G.P., Ertl, T., Nielson, G.M.: Scientific Visualization: The Visual Extraction of Knowledge from Data. In: Mathematics+Visualization. Springer, Heidelberg (2005)
15. Blazona, B., Mihajlovic, Z.: Visualization service based on web services. In: 29th International Conference on Information Technology Interfaces, 2007. ITI 2007, June 25-28, 2007, pp. 673–678 (2007)
16. Brusilovsky, P., Kobsa, A., Nejdl, W.E.: The Adaptive Web (2007)
17. Eirinaki, M., Vazirgiannis, M.: Web mining for web personalization. ACM Trans. Inter. Tech. 3, 1–27 (2003)
18. Auber, D.: Tulip: A huge graph visualisation framework. In: Mutzel, P., Jünger, M. (eds.) Graph Drawing Softwares. Mathematics and Visualization, pp. 105–126. Springer, Heidelberg (2003)