POLITECNICO DI MILANO
Scuola di Ingegneria dell'Informazione

POLO TERRITORIALE DI COMO

Master of Science in Computer Engineering

Design and Implementation of an Interactive Data-driven Web-based Visualization Framework for Sense-making of Elderly Daily Life

Supervisor: Prof. Sara Comai
Assistant Supervisor: Hassan Saidinejad

Master Graduation Thesis by: Diego Bogni
Student ID Number: 801144

Academic Year 2013/14

POLITECNICO DI MILANO
Scuola di Ingegneria dell'Informazione

POLO TERRITORIALE DI COMO

Corso di Laurea Magistrale in Ingegneria Informatica

Progettazione e Sviluppo di un Framework di Visualizzazione Interattivo Data-driven e Web-based per l'Esplorazione del Comportamento Quotidiano di Persone Anziane

Relatore: Prof. Sara Comai
Correlatore: Hassan Saidinejad

Tesi di laurea di: Diego Bogni
Matricola: 801144

Anno Accademico 2013/14

Sommario

Negli ultimi decenni le aspettative di vita della popolazione si sono notevolmente allungate. Questo andamento, che non intende rallentare, ha portato ad un vistoso incremento del numero di persone anziane facendo sorgere alcune problematiche di natura socio-economica e assistenziale causate dal naturale processo di invecchiamento dell'essere umano. La necessità, quindi, è quella di garantire un soddisfacente livello di assistenza e benessere alle persone anziane. L'attuale modello di assistenza non è adeguato per garantire tale benessere ad un vasto numero di pazienti. Inoltre, molto spesso queste soluzioni non sono accettate dall'anziano, che nella maggior parte dei casi preferirebbe continuare ad avere una vita indipendente nella propria abitazione, evitando di dover subire cambiamenti radicali. Lo spostamento in case di riposo comporta alti costi e in molti casi un declino nel benessere del paziente. Sfruttando le potenzialità dell'Information and Communication Technology è possibile ridurre i costi di assistenza e ritardare il trasferimento in case di riposo, garantendo un adeguato livello di benessere. Per supportare e migliorare la vita indipendente di un anziano, un approccio molto utilizzato è Ambient Assisted Living (AAL). Installando nell'abitazione dell'anziano una serie di sensori, attuatori e interfacce di comunicazione, è possibile trasformarla in una smart-home, senza richiedere particolari cambiamenti nello stile di vita. Tale configurazione permette di raccogliere un enorme quantitativo di dati, dando la possibilità di poter studiare ed analizzare le abitudini di vita dell'anziano. Questo aspetto però, crea i presupposti per affrontare una nuova sfida: estrarre informazioni significative dai dati. Una possibile soluzione a questo problema è la realizzazione di un framework di visualizzazione focalizzato sulla presentazione di dati attraverso una serie di visualizzazioni, che stimolino la percezione dell'utente (solitamente persone che si prendono cura dell'anziano) e, anche attraverso dei meccanismi di interazione, permettano di ottenere una corretta e significativa interpretazione dei dati, al fine di poter garantire una adeguata assistenza sanitaria. In questo lavoro è stato realizzato un framework di visualizzazione composto da due tipi di presentazioni: Layered Aggregate Radial Tree (LART) e Rich State Transition Graph (RSTG). Seguendo le più diffuse teorie sulla visualizzazione e sulla percezione visiva, le due visualizzazioni proposte hanno come obiettivo quello di mostrare in maniera chiara ed intuitiva i dati raccolti dai sensori in un sistema AAL, favorendo l'esplorazione dei dati e l'estrazione di informazioni rilevanti.

Abstract

In the recent years, the expected lifespan has got longer with respect to the previous decades. This trend has led to a considerable increase of the number of aged people involving socioeconomic and health-related complications due to the ageing process. Therefore, preparation is needed to confront this emerging challenge: the well-being of the aged people. Current healthcare system has no enough resources in order to ensure a satisfactory assistance level for a large number of elderly. Moreover, seniors often do not consider these solutions acceptable, since one aspect of well-being is to enjoy an independent life in their own familiar environment avoiding radical changes. Early institutionalization involves high costs and in many situations a decline in the patient's well-being. Exploiting the potentialities of Information and Communication Technology could reduce care costs, delay the need for institutionalization, granting a satisfactory well-being level. Ambient Assisted Living (AAL) aims at promoting and supporting an independent life for the elderly people. It consists in the transformation of the older adult's dwelling into a smart environment by installing sensors, actuators, and interfacing mechanisms, in order to monitor the resident's life in an unobtrusive way. In such a setting, a huge amount of data, from various sources, could be collected, providing the opportunity to explore and analyze the daily life of the resident. At the same time, having such amount of data introduces a new challenge: the extraction of meaningful information in order to make sense of it. A possible solution is the development of a visualization framework focused on the presentation of data through visualizations that stimulate the human perception and, with the help of interactive mechanisms, makes possible to gain a correct and significative data interpretation, ensuring an adequate healthcare. In this work a visualization framework that is composed of two visualizations is proposed: Layered Aggregated Radial Tree (LART) and Rich State Transition Graph (RSTG). Following the most prevailing theories about visualization and human visual perception, the proposed visualizations aim to clearly and intuitively illustrate data gathered from an AAL system, promoting data exploration and information seeking.

ACKNOWLEDGMENT

I would like to thank my supervisor, Professor Sara Comai, for her precious advices and supervision.

I wish to express my sincere gratitude to my colleague, Hassan Saidinejad, for his encouragement, patience, and inestimable assistance in the development of this work.

Last but not least, I thank my family and all my closest friends for their continuous support during my academic path, and their blind trust in me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Assistive Technology Group

The Assistive Technology Group (ATG) is a multi-disciplinary working group of Politecnico di Milano, whose mission is to "Identify, forecast, implement, promote and apply, innovative method and technology from ICT for the development of sustainable solutions for frailty (disabled, elderly, difficulties) which guarantees the recovery of functionality, social integration, equal opportunities, health, self determination and quality of life". ATG gathers a set of competences from ICT, Design and Management acting on a quality of life projects; these set of competences is transversal to the considered mission creating, in this way, a research and innovation-maker structure flexible and adaptive. ATGroup faces with needs that, currently, have no commercial answers such as analysis, identification and design of innovative solutions in the field of ICT (e.g. graphical applications, efficient and low power wireless transmission, natural language processing, and high accuracy indoor localization). A way for supporting independent life consists in monitoring and analyzing the life style of the assisted people, in order to realize when anomalies occur, and to support the analysis of their causes. A branch of ATG focuses on making sense and insight of data through data analysis and visualization techniques as a tool for helping independent ageing, as well as user interface and interaction aspects.

1.2 Problem Statement

The number of elderly people is continuously increasing since the expected lifespan is getting longer than in previous decades [1]. It is accepted that our health services cannot continue to provide hospital-based care to this segment of population. This will be a problem in the next years, since the existing resources for providing care to the elderly are inadequate to support their essential health and social needs.

The ageing process usually involves a loss of mental and physical abilities which raise the risks of accidents and the probability of making wrong actions. Often the senior is no more able to have an independent life with a minimum standard of quality. Nowadays, when someone is in this situation, there are few ways to overcome this problem: (i) to be admitted to a nursing home; (ii) to be cared by an in-home nurse; (iii) to relocate into the dwelling of the people in charge of the person. Unfortunately, these solutions are not always feasible because they usually require an important economic effort, and is not always possible to find a solution that fits the needs of all the people involved (e.g., can be difficult to find a nursing home place in the neighborhood of the people in charge of the senior). Moreover, often these options are not accepted by the seniors that would prefer to continue living in their usual environment without suffer too many changes in their life style. When this not happens, it leads to demoralization and thus a decline in their general well-being.

The usage of technology could be beneficial for the independent living of the elderly. The maturity level and the pervasive use of telecommunication devices make possible the development and improvement of assisted living. A widespread approach consists in upgrading the residents dwelling in a smart environment, by installing specific sensors, actuators, communication hubs and interfaces. These Ambient Assisted Living (AAL) services provide information about activity-sequence-awareness, location-awareness, presence-awareness, and context-awareness capabilities that can be processed and analyzed by experts.

Such an AAL system and the information gathered from it can augment the patient care practices in several ways: monitoring the person's activity in real-time interacting with him/her, or alerting carers in case of a harmful event. Furthermore, logs of the patient's activity, annotated with the sensor readings, can be processed and analyzed by software in order to study patients personal behavior and to detect person-specific behavior anomalies which may indicate a disorder in the residents health.

In such a setting, a huge amount of data from various sources could be potentially collected: health-related data such as daily blood pressure, daily activity levels, and heart rate; environment-related data such as the temperature, humidity, and luminosity; person-environment-related data such as interaction with home apparatus and presence in specific places of the home at specific times, just to name a few. Having such amount of data provides the opportunity to explore and analyze the daily life of the older adult. Data in raw format are not suited for being analyzed by humans. In that representation, important information cannot be extracted, remaining "hidden" to use user's eyes. This fact implies a loss of relevant information. Representing data in visual form makes it possible to discover the whole information set contained in data. Thus, developing meaningful data visualizations can highly improve the end-user's capabilities to make sense of data, taking better decisions, and wasting less time. Since the range of end-user of these kinds of systems may vary from domain-specific experts, who are interested in investigate a phenomenon (e.g., doctors, specialists), to caregivers (or patients) who typically are more interested in the person's well-being, the relevant information vary too. Generally speaking, each group of end-users has different information needs. This means that the developed visualizations must be reliable and flexible, in order to be adapted to the different user's needs. Moreover, the quality of visualization can determine the quality of the end-user decisions and performances.

So, the main goal of this work is to support the independent life of aged people, exploiting the advantages offered by AAL systems. The possibility of monitoring and analyzing the resident's life style can reduce the assistance costs and increase the well-being of the elderly. AAL systems make it possible to collect a huge amount of data. Extracting the relevant information from this data permits the end-user to discover important facts about the aged person. Visual data representation increases the possibility to retrieve these information, making it possible to improve the monitoring and assistance capabilities. People in charge of the aged person can understand, from remote and in asynchronous way, the general behavior of the assisted person and realize if significative variations occurred.

1.3 Visual Analytics

Visual analytics is defined as the science of analytical reasoning facilitated by interactive visual interfaces [2]. In a less formal way, it can be considered as a form of query aimed to solve a problem, where data are expressed in a graphical and interactive manner. In particular, the main goal of visual analytics is to provide insight about the proposed problem.

Nowadays, a huge amount of data is produced at an extremely fast rate, and the technologies for collecting and storing the data are at a later stage with respect to the data analysis capabilities. In the past years, many different methods for analyzing data have been developed, but the intrinsic complexity of many problems, ensures that human abilities must be taken into account at the early stages of the data analysis process.

Analysts, decision makers, and many others professional figures, depend on information contained inside the data. One of the most important challenges of today is to meaningfully make use of the immense amount of data available. This information overload can bring the decision makers to get lost in irrelevant or misleading information, reaching null or wrong conclusions. The efficient use of data, instead, can help realizing technological progress and business success.

Therefore, visual analytics can be seen as an approach combining visualization, human abilities, and data analysis. Gain insight into complex problems can be done by mixing the computers computational and the storage capabilities with the humans background knowledge, and flexibility. The visualization part has particular importance, since it is the joining link between data and human. It aims at communicating the information contained in the data to the user, stimulating the cognition and perception mechanisms of the human. This has a very important role in a decision-making process.

Visual analytics is helpful in applications involving large, complex datasets and analytical processes that require a high degree of monitoring and interaction (i.e., data mining uses). The approach uses data visualization technologies to help users identify trends, patterns and relationships in the data they are working with. Visual analytics tools make it easier for non-technical users too. Visualizations allow the user to directly interact with the analysis methods by adjusting parameters or selecting other algorithms of analysis. The result can be continuously refined and verified. Misleading results in an intermediate step can thus be discovered at an early stage, leading to better results and a higher confidence. All these features make visual analytics an interesting field of research.

1.4 Objectives of the Thesis

The main contributions of the thesis are:

- A review of the state of the art for data and AAL systems visualizations;

- A review of the human visual perception mechanism, and how data visualization can take advantage from it;
- The design and the implementation of a data visualization framework composed of two complementary visualizations aimed to facilitate the exploring and information-seeking tasks in AAL systems;
- An evaluation process designed to understand the usability of both the visualizations and possible improvements to the framework.

1.5 Structure of the Thesis

The rest of this work is organized as follows. Chapter 2 is dedicated to the presentation of the related works in the fields of AAL systems and data visualization. Chapter 3 illustrates the most common visualization and information-seeking theories and the methodology applied in this work. The design and implementation of a visualization framework aimed to elderly's life style analysis is detailed in Chapter 4. Chapter 5 describes an implemented demo, and it analyzes the results of an usability evaluation process. Finally, Chapter 6 discusses the results obtained in this work and indicates which are the possible improvements to this work.

Chapter 2

Related Works

This chapter illustrates the related works in the field of data visualization. In particular, it presents works regarding event-based and generic life monitoring visualizations, followed by visualizations specifically designed for presenting AAL systems data.

2.1 Data Visualization

In [3], the authors show an example of diary of data collection where, through a mobile application, users can record their daily activities. Each diary represents a single day and it is visualized as a vertical bar composed of the sequence of performed activities. Color represents activity category, while time is represented on the y-axis. The visualization of a collection of diaries is done by putting side by side each singular diary (see Figure 2.1 for an example).

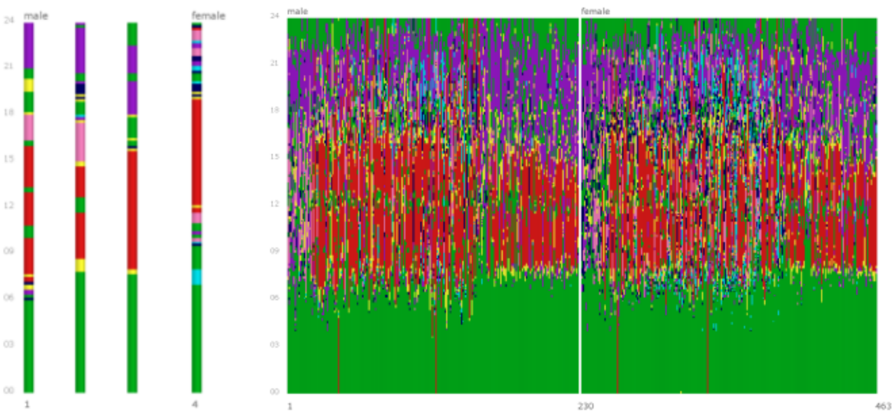


Figure 2.1 Visual representations of activity diaries [3].

Based on activity diaries, ActiviTree [4] has been proposed as a meaningful way to represent sequence of events in presence of a large dataset. The proposed algorithm, for the systematic identification of sequences in activity diary data, is based on web searching algorithms using a matrix similarity and assigning similarity scores to the events which depend both on connectivity and frequency. These weighting scores is used to rank each sequence during user analysis. In an interactive way, the user starts selecting an activity and all the preceding and subsequent activities are sorted by rank importance. Then, the user can select another connected activity creating in this way a path and the process can be iterated. Figure 2.2 depicts an example, in which every stage permits the user to see also the distribution, time of occurrence and duration of the selected sequence.

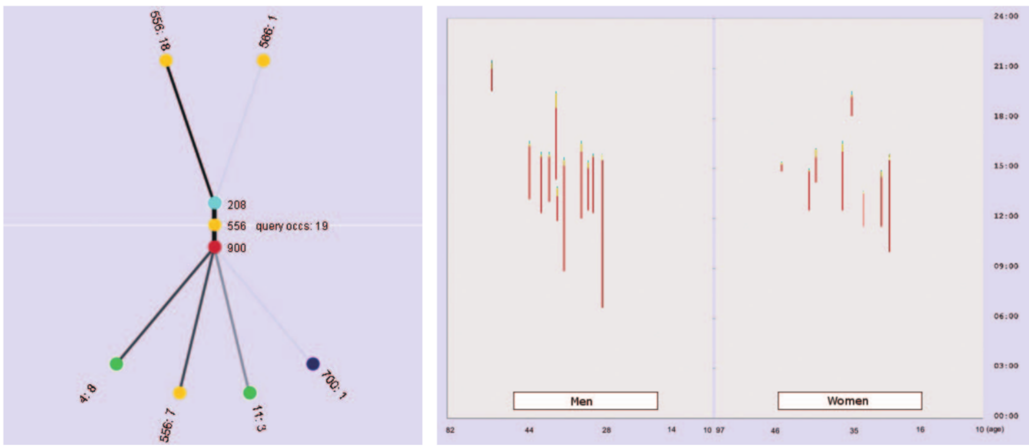


Figure 2.2 Example of an exploration sequence. On the left, ActiviTree. On the right, is shown the distribution of the explored sequence [4].

Since daily life activities are strongly related with time, the visualization of time-series data can be meaningfully done with spirals [5]. The analysis of time-series permits to discover trends, periodic behaviors and to predict future developments. Usually time-series dataset are represented with line graphs or bar charts but they do not fit very well when the amount of data increases. The spiral visualization takes advantage from the human ability to detect structures and patterns making it possible to identify periodic structures in the dataset.

An example of visualization of personal histories is LifeLines [6]. It is a general visualization environment that can be applied to medical and court records and other types of biographical data. It shows an overview screen where each record is represented by a line of different color and thickness, depending on the records features. In this visualization all the records have a time-component that is easily reported on the x-axis of the visualization. The main benefits of such approach are the reduced probability of missing information and the facility to spot anomalies and trends. The former benefit is given by the intrinsic structure of this framework, since it presents an overview for the given subject in a single screen with the possibility to go deeper in details when needed. The latter is based on the fact that it is easier to estimate intervals displayed as distances between segments with respect to raw dates displayed in a table: cycles, trends and anomalies are quickly detectable. Wang et al. developed an extension of LifeLines in [7]. Lifelines2 is an interactive visual tool that supports the visual exploration of multiple records of categorical temporal data. In the medical field, displaying the patient histories aligned on sentinel events enable medical researchers to spot precursor, co-occurring, and aftereffect events. Each record that matches an input query is vertically stacked in alternating background color. Events are represented by colored triangles and the display is fitted so that the entire date range fits in the screen. A series of controls permit the user to filter, align and rank the data. The event sequence analysis is an important task in medical domain, but in many cases medical researchers have to deal with thousands of records. LifeFlow [8] is an interactive visual overview of event sequences. It is able to summarize all possible sequence and represents the temporal spacing of the events within sequences. In LifeFlow all records are aggregated into a tree-based data structure, where they are grouped event-by-event from the beginning of the event sequence to the end. Then, each node of the tree is represented with a color-coded event bar, matching the color of the event type. The height of a bar is determined by the number of records in that node proportionally to the total number of records. The horizontal gap between a bar and its parent is proportional to the mean time between the two events. LifeFlow can be combined to Lifelines2 facilitating the data exploration by allowing users to review individual records (see Figure 2.4 for an example).

Timeline Tree approach is presented in [9]. It is an example of visualization of sequences of transactions in information hierarchies. In a single diagram TimeLine Tree integrates three views (see Figure 2.5). Starting from the left, the “Information Hierarchy” interactive view shows the entire hierarchy. It is possible to expand or collapse the hierarchy by clicking on a node. Then, the “Thumbnails” view is a small representation of the “Timeline” view. The latter is the visual representation of the sequence of transactions. “Timeline” provides a way to explore and analyzed the transaction.

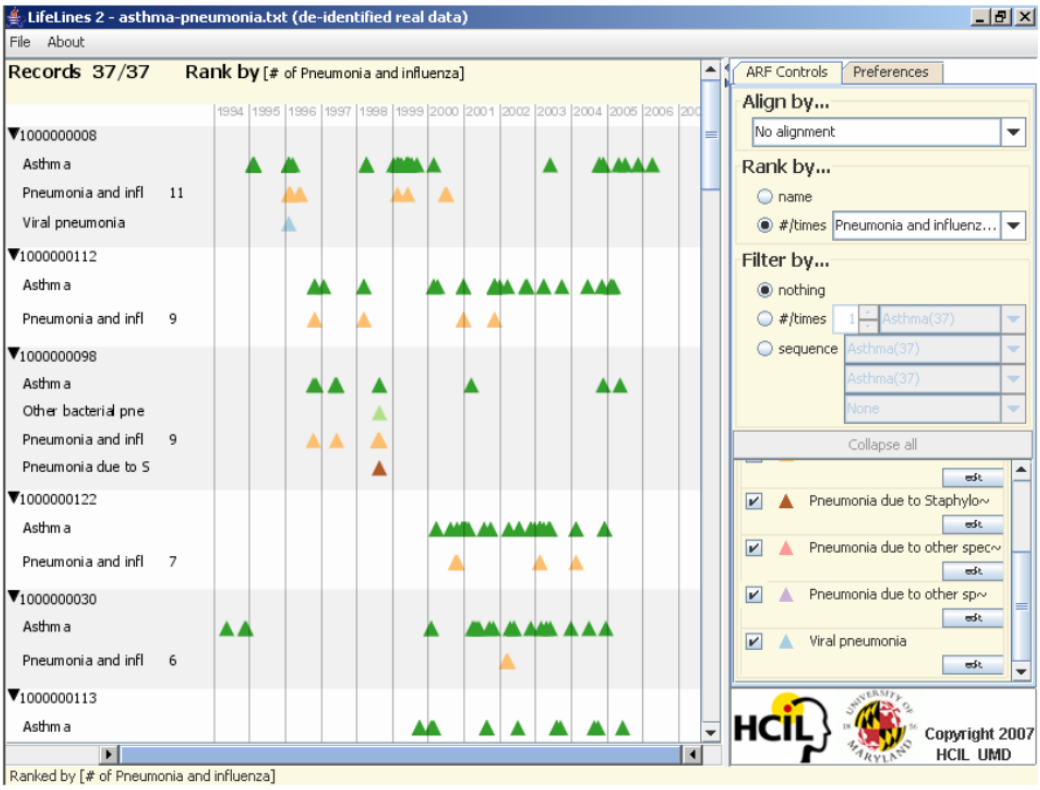


Figure 2.3 Interface of Lifelines2 [7].

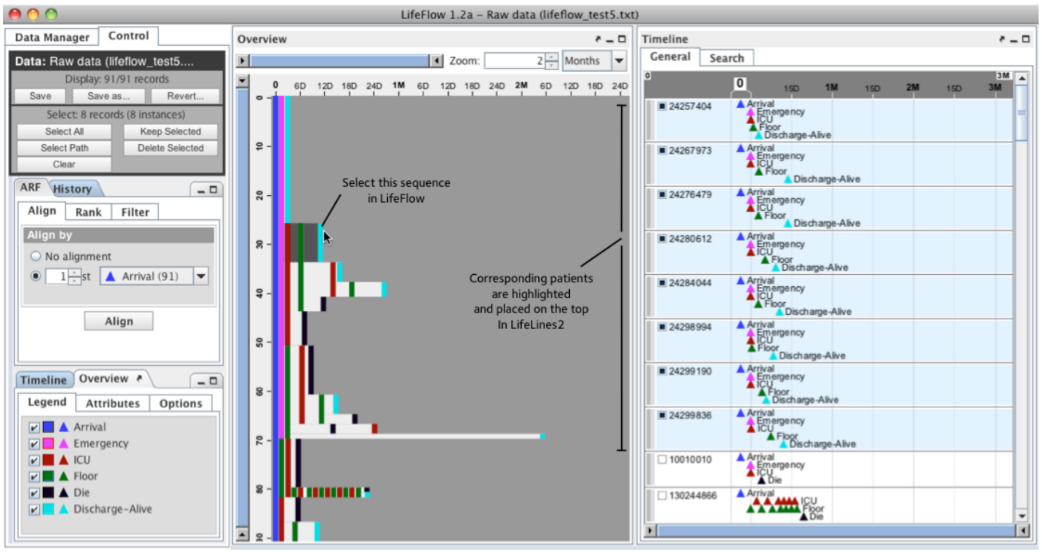


Figure 2.4 Example of LifeFlow used in combination with Lifelines2 [8].

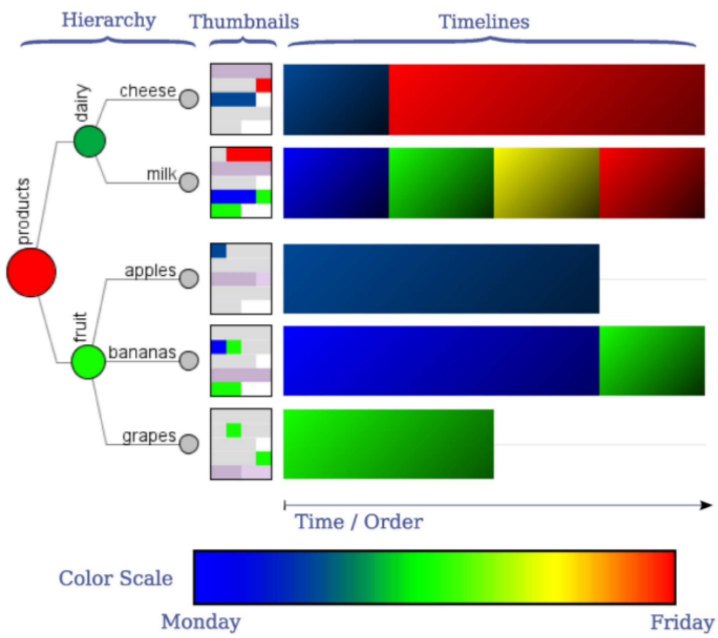


Figure 2.5 Example of TimeLine Tree [9].

2.2 Data Visualization for AAL Systems

The amount of data collected by AAL systems requires visual tools in order to represent in a meaningful way the activities performed by the resident of the dwelling. In the fields of elderly monitoring, independent living, and behavioral pattern recognition many works has been done. One of the first examples that makes use of AAL system data is [10], where Virone et al. presented a method for discovering behavioral patterns of elderlies in smart home environments. With the help of a pattern mining software (SAMCAD), this method defines behavioral patterns and, by computing some norms and deviations from the norms, it discerns normal behaviors and not-normal behaviors. People in charge of the resident are alerted in case of alarmingly behavior.

In [11], authors discussed the needs of visual presentation of AAL data by different group of users. It reported many commercial and academic examples highlighting the fact that it is necessary to develop interfaces and data visualization components that can communicate clearly the health status of the care recipients to different user groups. Figure 2.6 shows an example of the application that uses life patterns for providing enhanced home-based care. It is taken from [12], where the day life patterns of individual have modeled. Key concept in this work is “busyness”, defined as the measure of overall movements, presence in locations and interaction with objects inside a smart home. The target of this work was to identify behavioral patterns in the daily life of residents. Starting from the data collected by the sensors, they had been visualized after summarization, aggregation at different levels of granularity and trend analysis.

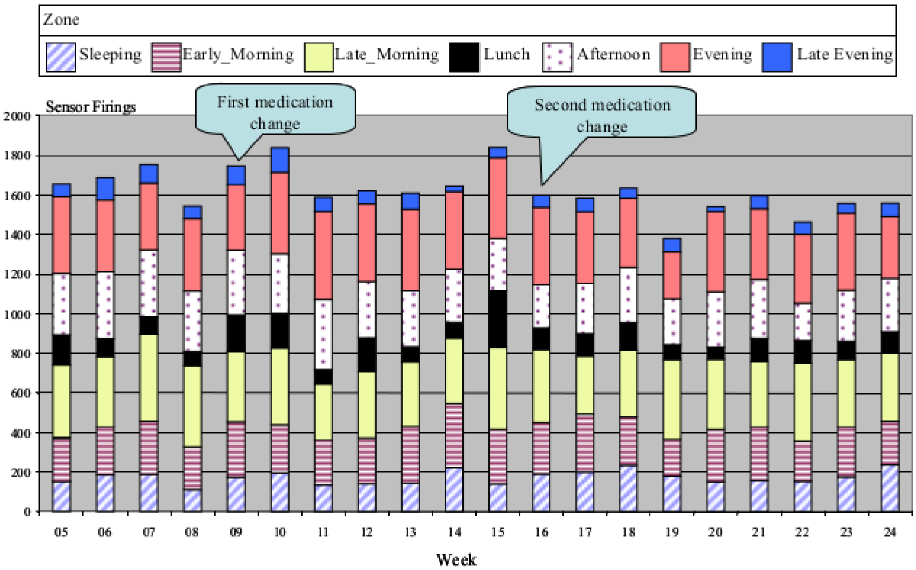


Figure 2.6 Example of home-based care application showing busyness by time zone [12].

Robben et al. expose the result of the interview to different types of elderly care specialists to gather opinions and needs about sensor data visualization [13]. It emerged that it is very important to start from activities of daily life pattern and be able to catch significant deviations from it. In this way caregivers can be aware of an impending disease and adapt their treatments. Since decisions must be taken as soon as possible, Gestalt laws has been used to express the information in the most relevant way. Figure 2.7 shows the final version of the application, approved by the interview specialists. The possibility to see intra-day variation and, most important, per-day variation are the most appreciated features of the visualization. Furthermore, activity description by color help to have an insight about relevant behavior deviation, although specialists claimed that a more evident signaling of deviations would be very appreciated. Zooming and selection tools are considered strong help for discarding irrelevant data (i.e. particular time intervals) and get a

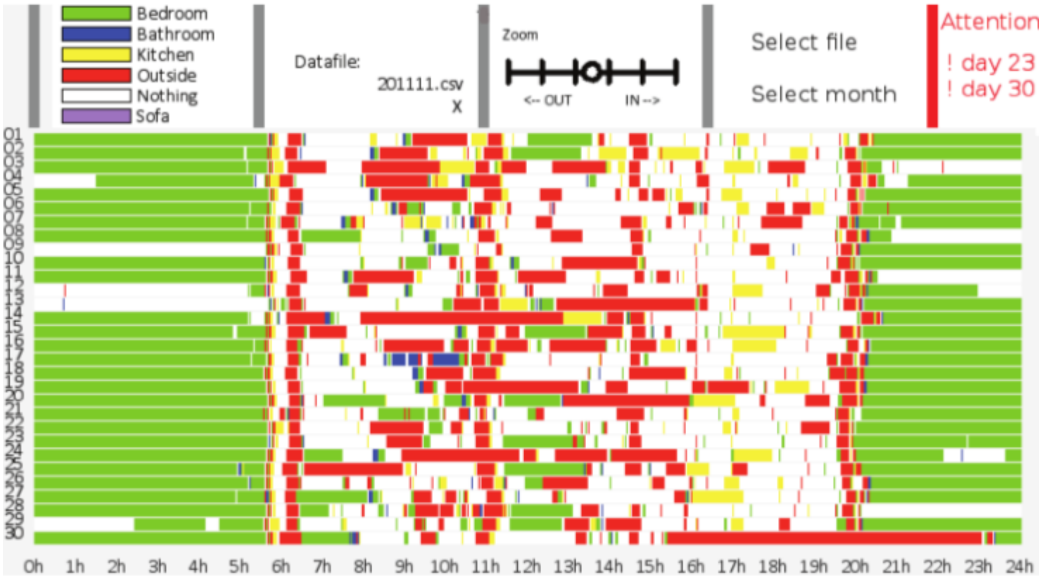


Figure 2.7 An example of the application developed in [13].

deeper knowledge.

Another example of visualization of day life activities is presented in [14]. It proposes the VISUAL-TimePAcTS framework as a visual-analysis tool for sequential pattern identification. The authors state that traditional time-use studies do not take into account important information like the place and the context in which the activities are executed, the frequency and the duration of each activity. These hidden information are very important for discovering and deeply understanding daily life patterns. Once patterns are identified and formatted as n-tuples, they are displayed in a "default" visualization. Users can select a subset of patterns and these will be highlighted with the activity path that contains them. In this way the selected patterns can be studied in the context of the visualized individuals' day without the distraction of the surrounding activities. Figure 2.8 shows an example of the identification of the pattern for the single activity *eat dinner* performed by the VISUAL-TimePAcTS framework.

The previous mentioned works do not take into account the fact that two or more activities of daily living can be done simultaneously. In [15], authors propose an approach for analyzing patterns in activities daily living as well as spatio-temporal relations. They determined the probability relations between activities that occur at the same time, for extracting simultaneous activities patterns incrementally. Furthermore, they proposed a visualization

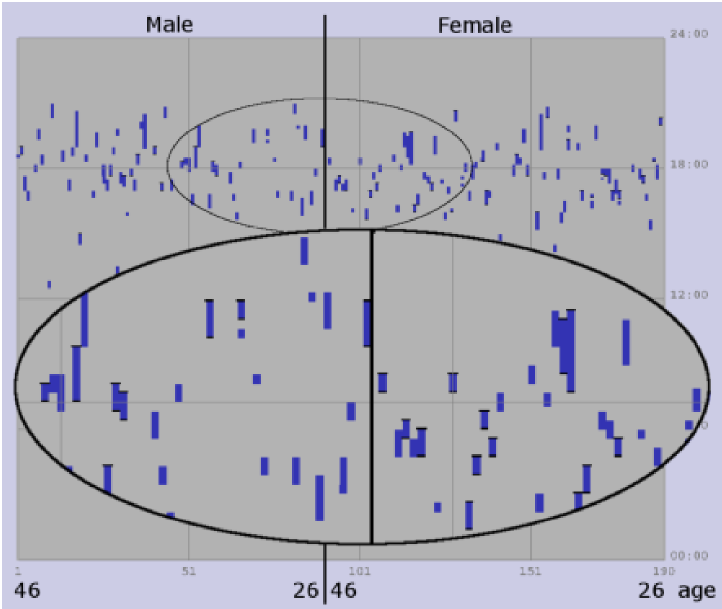


Figure 2.8 Example of patterns identified by the VISUAL-TimePAcTS framework [14].

framework based on the frequency of relation between activities, and the percentage of sequential relations from accumulated frequency. The activities set (or a single and non-simultaneous activity) are represented by circle, in which the radius is proportional to the relative value of duration and the brightness represents the relation value of frequency (see Figure 2.9 for an example).

In [16], the authors used a density map to analyze the data extracted from sensors logs in the homes of seniors (see Figure 2.10). Each line of the density map represents a day, sampled by hours. The y-axis represents days in a month. The color of each cell represents the density of events raised by the sensors and it ranges from white (no sensors activated) to blue (550 events per hour). An exception is the black color assigned to the time spent away from home that has an accuracy of seconds. Starting from that, Wangs et al. introduced the concept of dissimilarity between density maps as a method for detecting changes in the patterns of residents that aids caregivers in the monitoring process [17]. The maps dissimilarity measure is based on the co-occurrence matrix. The daily life pattern can be extracted from the textural features (e.g., spatial and frequency properties) and structural features (e.g., coarseness and periodicity) of the density maps. The dissimilarity of two different density maps is represented by the distance from one map to another

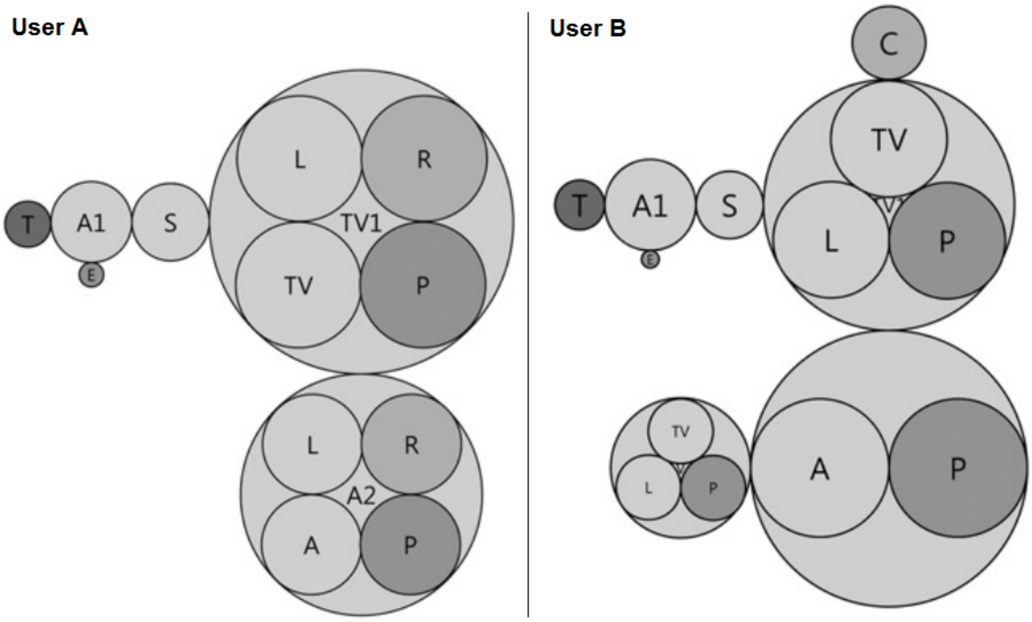


Figure 2.9 Visualization for User A and User B (E: entrance; T: toilet; A: audio; S: shower; L: laptop; R: reading; P: phone; C: coffeepot) [15].

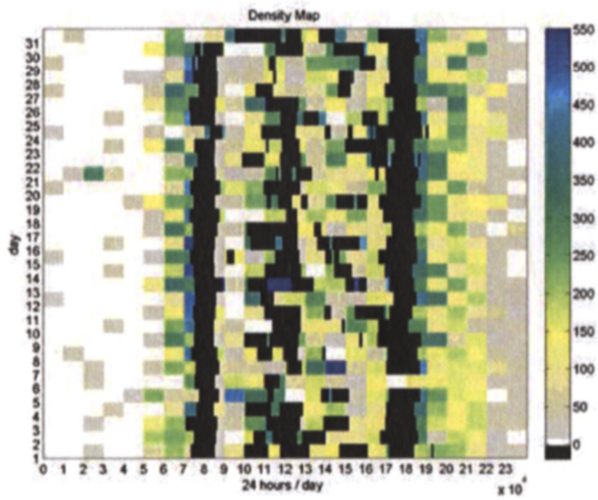


Figure 2.10 Example of a density map showing an active life-style [16].

computed in feature space: the smaller the distance, the more similar the density maps are. Empirical results have shown the normalized Euclidean distance to be the most sensitive and, thus, was chosen for the dissimilarity measure.

A framework for reliable and non-intrusive home assistance is described in [18]. Events recorded by sensors connected to a Wireless Sensor Network are stored and processed by a web service that shows off REST APIs for visualizing data in a variety of views. One of them represents the three-dimensional model of the Smart Condo space augmenting the degree of realism and intuitiveness. The system, that can approximate the residents position and actions, is useful for monitoring seniors activity in real-time (i.e., alerting caregivers in case of harmful events) and for real-time interaction between the resident and caregivers.

[19] shows how it can be easy to upgrade a normal apartment into a smart home. The applications included in the kit process and analyze the data collected by the sensors, performing activity recognition and activity discovery tasks (working on unlabeled data). On top of this work, two visualizations [20], [21] have been created. PyViz is a desktop visualization, while CASASViz is web oriented and developed for working also on smartphones. Both allow the user to quickly check updates as the sensor environment evolves. CASASViz offers seven different visualization applications: Main Visualizer indicates the location of the resident in the house (see Figure 2.11b), Mobility Heat Map (see Figure 2.12a) describes the frequency of the sensor events triggered by the residents. Activity Graph (see Figure 2.12b), Power Usage Visualizer, Long-term and Abnormal Patterns Visualizer, and Activity Feature Extraction make it possible to identify behavior patterns taking into account different aspects of the resident's life.

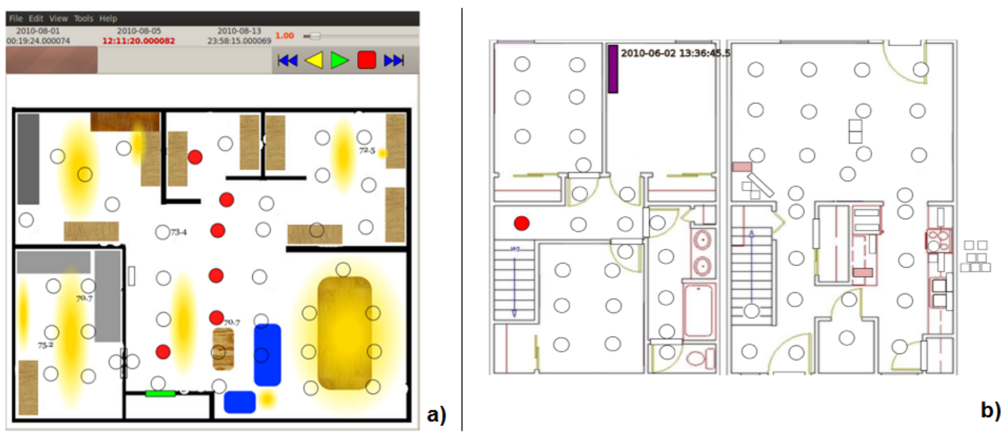


Figure 2.11 Examples of PyViz [20] **(a)**, and CASASViz Main Visualizer Interface [21] **(b)**.

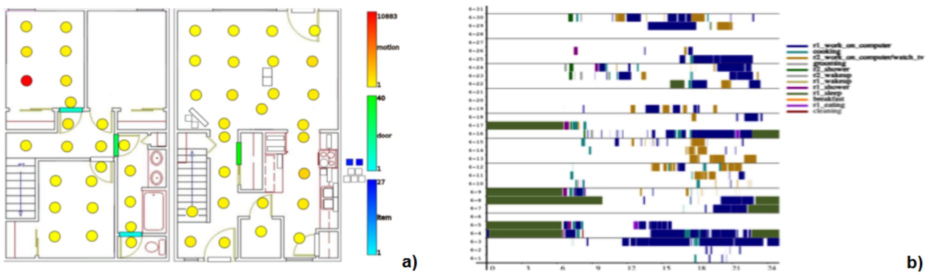


Figure 2.12 Examples of CASASViz Mobility Heat Map **(a)** and Activity Graph **(b)** [21].

Chapter 3

Methodology

In this chapter the common visualization theories and the adopted methodology are described. The first part reports the concepts of human visual perception, cognition, and a brief explanation of how they work in the human brain. Then, how data visualization and information-seeking can take advantage from these mechanisms is analyzed. In the second part, three visualization frameworks are presented. They explain different ways to reach efficient visual data extraction. Finally, on the basis of the mentioned visualization theories, the main line followed in this work is reported.

3.1 Visual Perception and Cognition

It happens frequently to find terms like data, information and knowledge in visualization jargon. In many cases they are used as synonyms, sometimes they indicate different levels of abstraction. Table 3.1 shows Ackoff's definition of these terms in the perceptual and cognitive space.

Category	**Definition**
Data	Symbols
Information	Data that are processed to be useful, providing answers to "who", "what", "where", and "when"" questions
Knowledge	Application of data and information, providing answers to "how" questions

Table 3.1 Russell Ackoff's definition of data, information, and knowledge in perceptual and cognitive space. [22, pp. 13]

The main goal of data visualization is to extract information and knowledge starting from data: thus, making sense of data. Data in textual format cannot communicate meaningful information about patterns, trends and exceptions in a satisfactory time. Instead, having data represented in a visual form, makes it possible to take advantage of the capacity of human brain, gathering valuable results, quickly.

The key to success of data visualization depends mainly on the degree of encoded information it communicates to the users and in the way their brain can extract this information and understand it. In other words, a good visualization must translate abstract information into a visual representation that can be easily, efficiently, and meaningfully decoded. There is no other way of presenting information so that structures, groups, and trends can be discovered among a huge dataset.

Data visualization is broadly used nowadays, but sometimes it is used in a wrong way. Many data representations are done with misleading elements (e.g., graphs, charts, tables) and/or attributes (e.g., colors, sizes, edges) that make them lose the great part of the visualization benefits.
Exploiting the advantages of data visualization requires to follow the principle derived from the studying and understanding of human perception. It must be aware that human brain is a powerful pattern-finding engine: if information structures can be mapped to easily perceivable patterns, then those structures will be more easily interpreted.
As Few [23] explained, data visualization is effective when it shifts the balance between perception and cognition on the perception side, exploiting all the human brain abilities: visual perception (seeing) is extremely fast and efficient. It is immediate and effortless. Cognition (thinking), instead, is much slower and less efficient.

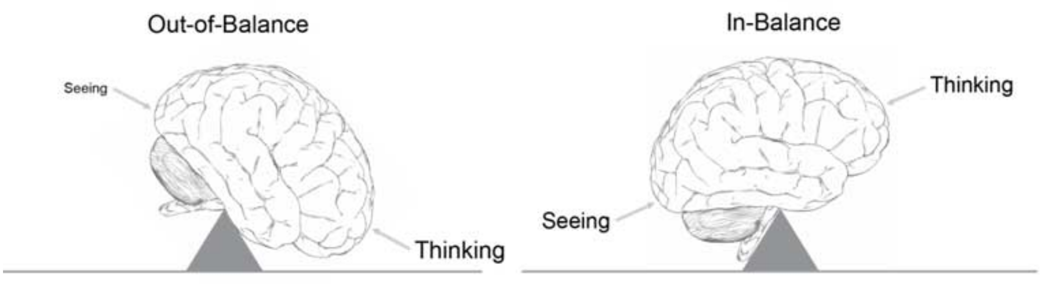


Figure 3.1 Perception and cognition balance [23].

We can roughly define two types of memory: working memory and long-term memory. Working memory is the one that holds the objects of immediate impact, while long-term memory contains the information we collect from everyday experience. They have to be considered as complementary, although in the visual process the capabilities and limitations of working memory are the most important and critical.

The strength of long-term memory is its flexibility. Studies suggest that we do not store such a huge amount of information about our life experiences, but instead, we reuse the existing knowledge adding only few details. This means that information is combined in many different ways and redundancy is highly minimized. Information located in long-term memory is retrieved through the chunking methods. A chunk can be anything like a mental representation of an object, a method for achieving some goals, or many other things. Thus, chunks representing simple concepts are grouped together producing more complex concepts.

Working memory can be subdivided in different subsystems for processing different types of input: visual, auditory information, body movements and many others. The visual working memory, the one activated by visual perception process, can be defined as the visual information gathered from what we are seeing: it contains position, shape, color and texture information of the visual scene. The problem is that only few simple objects can be retained. Usually this number varies from three to five, depending on the task and the pattern. This is the greatest limit of visual working memory. An example of this limitation is shown in Figure 3.2, where the theory of visual working memory suggests that three of the integrated glyphs could be held in visual working memory (left case), but only one of the nonintegrated glyphs (right case). This suggests that a perceptive visualization should prefer sensory symbols with respect to arbitrary. The former are very expressive and stimulate the visual sensory system, requiring no learning of their meaning. The latter, instead, must be interpreted and learned.

Another important aspect is attention. Most of the time we simply do not register what is going on in our environment unless we are looking for it. Although we are blind to many changes in our environment, some visual events are more likely to cause us to change attention than others are. The focus of attention largely determines what we will see, and this focus is set by the task we are undertaking.

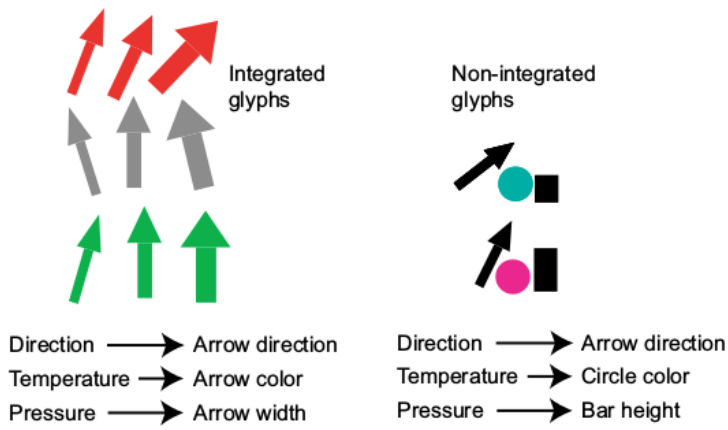


Figure 3.2 Comparison of multiple data attribute represented with glyphs [24, pp. 380].

Starting from these notions, the information extraction process for human visual perception can be expressed as a three stages model [24]. In the early visual processing stage, our brain extracts low-level features from the visual field (e.g., colors, textures, orientation of edges). All these information belongs to separated channels so, they are processed in parallel, rapidly. At this point, in the pattern perception stage, regions and simple patterns (e.g., regions of homogeneous color, with the same texture, or enclosed by contours) are discovered in a serial and slower way. Here, both the visual working memory and the long-term memory work. At the last stage, the visual working memory contains only the object demanded by active attention. At this level, only a few objects can be held at time. They are constructed from the available patterns that may provide answers to the visual query and from the information stored in long-term memory related to the task. For visual query it means a formulation of a hypothesis pertaining to a cognitive task that can be resolved by the discovering of a visual pattern. The patterns involved in visual problem solving are very different: pathfinding in graphs, quantity estimation, trend estimation, cluster identification, just for mentioning some.

Perceptual phenomena have been studied by a group of German psychologists of the Gestalt School of Psychology in the beginning of 1900s. Their results, the Gestalt laws of pattern perception are robust rules, accepted by unanimity, that describe the way humans see and percept the reality. The Gestalt laws have been easily translated into a set of design principles for information displays [24, Chapter 6]. The most important are:

- proximity: things that are close together are perceptually grouped together.

- similarity: similar elements tend to be grouped together.
- connectedness: connecting different graphical objects by lines is a powerful way of expressing a relationship between them (e.g., node-link diagram).
- continuity: we are more likely to construct visual entities out of visual elements that are smooth and continuous, rather than ones that contain abrupt changes in direction.
- closure: a closed contour tends to be seen as an object. Wherever a closed contour is seen, there is a very strong perceptual tendency to divide regions of space into inside or outside the contour. A region enclosed by a contour tends to group much stronger than simple proximity (e.g., Venn-Euler diagram).
- figure and ground: a figure is something object-like that is perceived as being in the foreground. The ground is whatever lies behind the figure. In general, smaller components of a pattern tend to be perceived as objects.

There are many other laws, but they are not of interest for the purpose of this work. These rules and other designing guidelines cannot be considered as dogmatic, but they may be applied in classical situations, enhancing the interpretability and self-explanatory features of a data visualization application.

3.2 Visualization Frameworks

It is not an easy task to acquire a thorough insight directly from a huge amount of data. Data visualization gives the opportunity to fill this gap. The visualization process is a function that maps the dataset to a visual representation which facilitates a more efficient and effective cognitive process for acquiring information and/or knowledge. In [22] Chen et al. claim that a typical visualization process is fundamentally the same as a typical search process. Given a dataset, the user makes decisions about which visualization tools to use for exploring the dataset, until he/she obtains a satisfactory collection of visualization results. The decision is taken by experimenting different controls such as styles, layout, colors, etc.

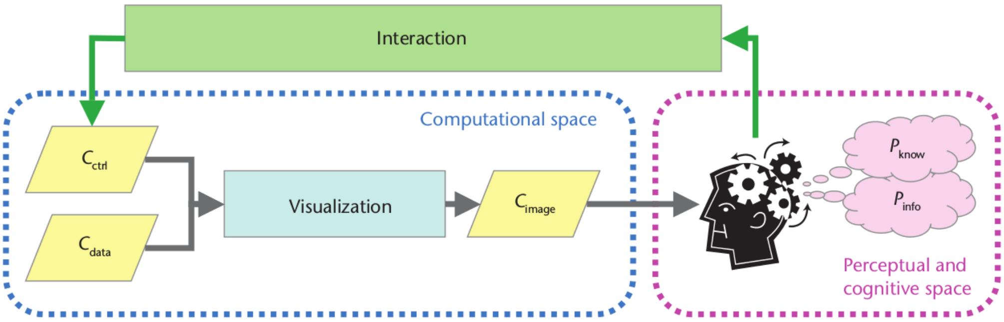


Figure 3.3 A typical visualization process [22].

In the visualization process the decision space is huge, since, for example, just changing the view position or trying out different transfer functions can produce very different results. With the aim of providing much more satisfactory outcomes in a quicker way, great importance has been given to user interaction. Interactivity can help in achieving this objective. However, working with very large dataset can slow down the visualization performance, and since the amount of data is incredibly increasing, interactive visualization alone results no longer adequate.

Information-assisted visualization expands the usual visualization process by adding information abstracted from the data. These information are typically about the input dataset, but they can also represent attributes of the visualization process or properties of the results. The user uses such information to reduce the search space. Examples of information are: geometric and temporal characteristics, topological properties, statistical indicators.

In Figure 3.3 and Figure 3.4 the interaction boxes sizes are intentionally different. This is due to the fact that in information-assisted visualization the degree of interactivity required, to reach the same results, is clearly lower with respect to a typical visualization process. The increasing size and complexity of data makes the help given by datasets information a necessity rather than an option.

The next step, is knowledge-assisted visualization. This approach considers the users knowledge as a fundamental factor in the visualization process. Sharing the domain knowledge among different users reduce the burden (to users) to acquire knowledge about complex visualizations, and enables the visualization community to learn and model the best practice. The knowledge representations can be given by expert users but in this way there may

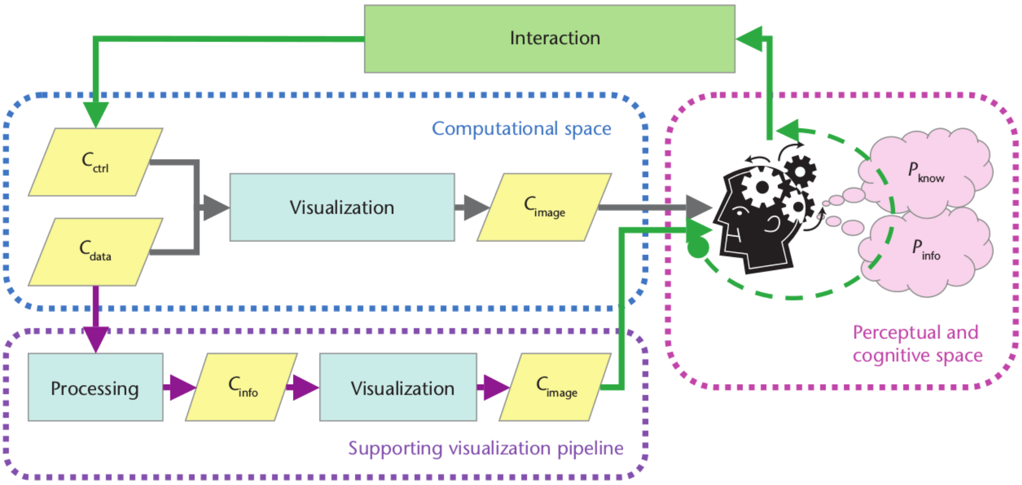


Figure 3.4 Information-assisted visualization [22].

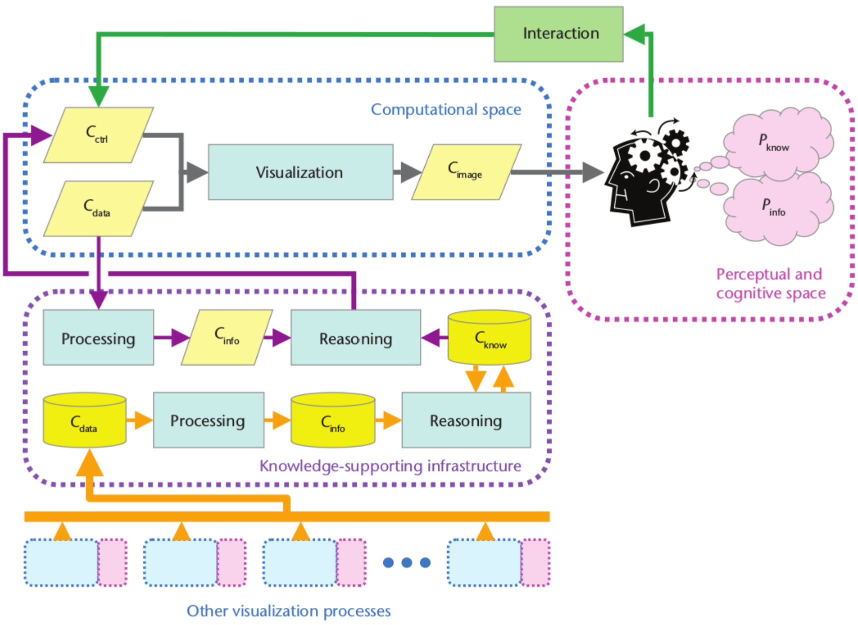


Figure 3.5 Knowledge-assisted visualization with simulated cognitive processing [22].

be difficulties in transcribing their knowledge. The solution can consist in creating a general purpose visualization infrastructure for collecting, processing and analyzing data about visualization processes in a systematical way. At the moment, knowledge-assisted visualization is at its early stages, while information-assisted visualization is evolving significantly from the stable and broadly-used interactive visualization approach.

3.3 Applied Method

This work follows the famous visual information-seeking mantra of "overview first, zoom and filter, then details-on-demand" [25] as general framework. It is accepted that this idea permits presenting information rapidly and allows for rapid user-controlled exploration. The mantra is composed of four tasks. "Overview" means to display an outline of the entire dataset, showing the general trend or a relevant subset of the most significant components. "Filtering" consists in hiding uninteresting items, letting the user focus on the most important items and dropping out the noisy ones. "Zooming" means focusing on a portion of the collection. "Details-on-demand" task gives details about a group or a single item, once it has been selected (and so required) by the user.

Two complementary visualizations have been developed in this work: Layered Aggregate Radial Tree (LART) and Rich State Transition Graph (RSTG). Both the visualizations are based on data collected by sensors activities in an AAL system, representing some aspects of smart-home resident's life.

LART displays information about the whole dataset in a compact way. It takes in input a hierarchical-structured dataset and shows the aggregated values at each level of hierarchy. This is the concept of overview. LART is designed mainly for temporal data, since a hierarchy can be easily done (e.g. years, seasons, months, days), but it can work also with non-temporal data extracting their intrinsic hierarchical relation. LART potentialities are enhanced by user interaction, which allows to zoom in at each level of the hierarchy and to select a portion of data to focus the attention on it (filtering). LART has many other features that will be deeply explained in Chapter 4.

RSTG performs the details-on-demand task. It is a variant of node-link diagram presenting information about the resident's behavior in a single day. For producing a reasonable and clear outcome, in LART some information are not representable. RSTG instead, focuses on those information expanding the ones visualized in LART. With the help of interactive tools, RSTG can perform further filtering operation. The complete feature-set of RSTG will be described in Chapter 4.

While LART visualization has a more generic target, RSTG is very related to the field of indoor spatial visualizations. According to the taxonomy given by Afyouni et al. in [26] about indoor spatial models, RSTG is a symbolic-based approach since the resident location is provided using conventional symbols and topological relationships (e.g., presence) are represented. Moreover, RSTG can be classified as graph-based, layout-based model, since nodes stand for places and each edge stands for the transition from a place to another. This kind of place-based graph is not suited for navigation purposes since the connection links between nodes do not represent a distance function but a frequency value. Instead, it is suited for location-awareness, activity-awareness and for spatial and behavioral analysis.

3.3.1 Guidelines

The visualizations developed in this work follow some of the guidelines listed in [24, Appendix C]. The aim of these recommendations is to suggest to the visualization designer the best practices in visual perception and to avoid misleading outcomes, resulting from collecting past empirical experiences. In [24], more than 150 design guidelines are proposed, spread through the wide domain of information visualization. Naturally, for this work, not all the guidelines have been followed, but the focus has been pointed to the subset including colors, contrast and lightness, pattern highlighting, interaction, and visual thinking.

The first suggestion is the one that can sum up the purpose of this work:

[G1.1] *"Design graphic representations of data by taking into account human sensory capabilities in such a way that important data elements and data patterns can be quickly perceived."*

The concept of perception is taken into account also in

[G1.2] *"Important data should be represented by graphical elements that are more visually distinct than those representing less important information."*

Concerning the general development of the visualizations:

[G6.20] *"Make every effort to standardize the mapping of data to visual patterns within and across applications."*

This is a way to reduce the quantity of background application information that the user needs to learn in order to exploit the whole features of the visualizations.

Another suggestion used to facilitate the user experience is derived from

[G6.5] *"Consider putting related information inside a closed contour. A line is adequate for regions having a simple shape. Color or texture can be used to define regions that have more complex shapes."*

Indeed, it has been decided to separate as much as possible the visualization part from the analysis tools. Particularly, in LART, but also in RTSG, the interactive tools (Selection and Similarity panels, see Chapter 4) are rendered in different areas, separated by a thin line which emphasize the different nature of the two displayed regions.

The design of the LART chart involves the followings, basic, recommendations:

[G10.5] *"Consider providing an overview map to speed up the acquisition of a mental map of the data space."*

[G10.6] *"Consider providing a small overview map to support navigation through a large data space."*

justified by the fact that the purpose of LART chart is to provide an overview of the entire dataset, displaying its general trend and the most salient phenomena. Also

[G8.18] *"When a large number of data points must be represented in a visualization, use symbols instead of words or pictorial icons."*

suggestion has been followed, because all the nodes (leaf and non-leaf) in LART can be abstracted to symbols or self-explaining objects.

In LART, labels have an important role in the exploration task. Following

[G8.19] *"Use words directly on the chart where the number of symbolic objects on each category is relatively few and where the space is available."*

[G8.20] *"Use Gestalt principles of proximity, connectedness, and common* *region to associate written labels with graphical elements."*

[G9.5] *"Place explanatory text as close as possible to the related parts of the diagram, and use a graphical linking method."*

LART's flexibility, gives the opportunity to decide which layers must have labels and which not. In the demo presented in Chapter 5, for example, it has been decided to avoid labels at leaf-nodes since that layer has a density of elements much higher than the non-leaf layers (see Figure 3.6).

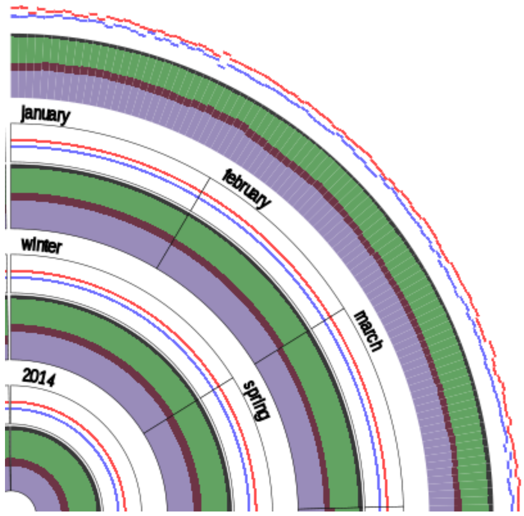


Figure 3.6 Example of LART label positioning.

In the Selection interaction it is possible to notice that the object representing the selected elements is an arc drawn with a saturated color, as suggested in

[G4.1] *"Use more saturated colors when color coding small symbols, thin lines, or other small areas. Use less saturated colors for coding large areas."*

[G4.16] *"Use low-saturation colors to color code large areas. Generally, light colors will be best because there is more room in color space in the high-lightness region than in the low-lightness region."*

Moreover, the default colors of the LART chart elements are less saturated, since their adjacency renders a big contiguous area.

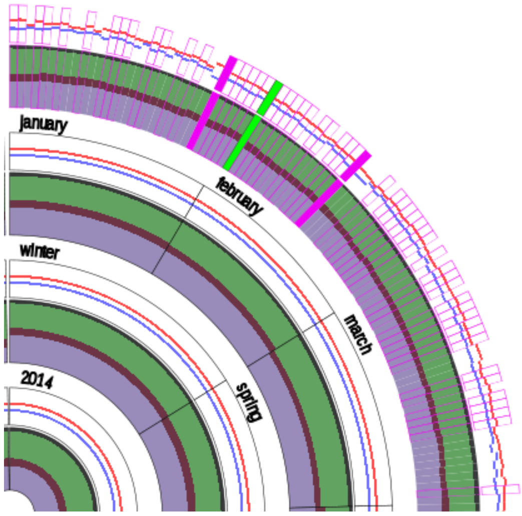


Figure 3.7 Example of LART Similarity interaction.

The Similarity interaction takes advantage from

[G5.7] *"For maximum popout, a symbol should be the only object in a display that is distinctive on a particular feature channel; for example, it might be the only item that is colored in a display where everything else is black and white."*

[G5.9] *"For highlighting, use whatever feature dimension is used least in other parts of the design."*

The selected element is highlighted, by default, with a very different color, and also matches are played up with colored borders (see Figure 3.7). This rules has been followed also in demo presented in Chapter 5, during the transition from the LART chart to the RSTG diagram. The selected day, indeed, is highlighted with a saturated color above the low-saturated colors of the LART layers (see Figure 3.8).

In the LART-RSTG transition

[G9.10] *"Use consistent representations from one part of a visualization sequence to the next. The same visual mapping of data must be preserved. [...]"*

[G9.11] *"Use graphical devices, such as frames and landmark object to help maintain visual continuity from one view of a data space to another."*

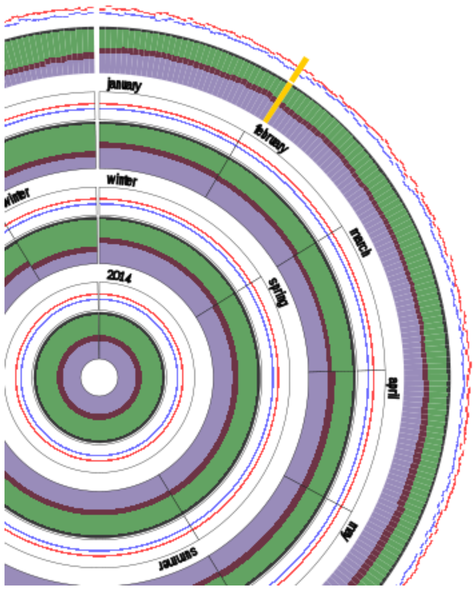


Figure 3.8 Example of LART to RSTG transition.

account for the small snapshot of the LART chart.

RSTG diagram covers great part of the previously mentioned guidelines. It also follows

> ***[G9.2]*** *"Graphical elements, rather than words, should be used to show structural relationships, such as link between entities and group of entities."*

Nodes are connected through links, which have also an important semantic value, and as explained in Chapter 6, future improvements can keep on following this rule.

Chapter 4

Design and Implementation

This chapter explains how the visualization framework for data related to AAL systems has been designed and implemented. The work follows as much as possible the theoretical principles and the guidelines expressed in Chapter 3.

The decision of developing a web application is based on the fact that in this way the problem of systems dependency will not be relevant anymore, making the implemented visualizations available for every web-accessible device. Web-oriented programming languages are having a great diffusion and this allows further improvements and manipulations of the framework. Moreover, in the last years many well-designed libraries concerning visualization have been implemented, permitting to achieve efficient results.

The framework implements two complementary data visualizations: Layered Aggregate Radial Tree (LART), and Rich State Transition Graph (RSTG). Both the visualizations are developed to represent a large amount of data and to support information-seeking in AAL systems. They are two types of interactive visualizations designed to stimulate the human visual perception. In particular, LART is a visualization that implements the "overview, zoom and filter" part of the visualization mantra. It provides an overview of the dataset and many techniques of interaction for focusing on a subset, and for discarding useless data. Exploration and analysis of the dataset are the main tasks offered by LART. RSTG implements the "details-on-demand" part of the visualization mantra. It provides a day-level detailed view of data and let the user get into the dynamics of day-level data.

4.1 Technologies

The framework is implemented in JavaScript using mainly the Data-Driven Document (`d3js.com`) library. Data-Driven Document (or D3) is a JavaScript library widely used in the field of data visualization. Web-based interactive visualizations usually combine different technologies: HTML for page content, CSS for aesthetics, JavaScript for interaction and SVG for vectorial graphics. Since all of them share the Document Object Model (DOM) representation of the page, it is easy to make them seamlessly work together. The main benefit of D3 is to selectively bind input data to arbitrary DOM elements, applying dynamic transforms to both generate and modify content: it supports data-driven manipulation of the document [27].

The basic operand of D3 is the *selection* that adopts the W3C Selector API to identify DOM elements. Elements can be selected by tag, class, unique identifier, attribute or position in the DOM hierarchy. Many *operators* can be applied to selected elements. The operators follows the W3C DOM API and make it possible to manipulate the selection content (e.g., attributes, styles, properties, and html). Operators values can be specified either as constants or functions, which are evaluated for each element of the underlying selection, and can be based on input data.

The `data` operator binds input data to selected nodes. It takes as argument an array containing the input data and then it computes the data join with the selected elements. Once the join is done, there are available the enter and exit subselections. The entering data have no corresponding nodes, while the exiting nodes have no corresponding data. The updating nodes are returned by the data operator. Figure 4.1 explains how the data join concept works. For example, a web application takes in input values regarding the

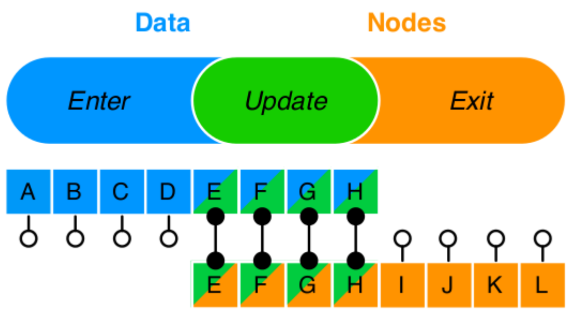


Figure 4.1 When new data (blue) are joined with old nodes (orange), three subselections result: enter, update and exit [27].

temperature of Como, sampled hourly. The application plots these data on a line chart, in real time, showing only ten values at time. Data are rendered as ‘circle’ SVG element of class ‘point’. When new data arrives a selection on the elements ‘circle’ of class ‘point’ is performed and the `data` operator is executed:

- the elements contained in the *update* subselection are shifted to the right by one position;
- a new ‘circle .point’ element is created, mapped to the incoming data, and plotted on the leftmost position of the chart;
- the “older” element, which is in the *exit* subselection, is removed.

The definition of *enter*, *update* and *exit* permits to control the entire element lifecycle. Static properties can be defined on enter, and dynamic properties on update. This is essential for dynamic visualizations.

User interaction is managed with the event handlers operators that, as DOM event listeners, execute callback functions receiving as parameters the user input targeted at specific elements and the element associated data, permitting the realization of data-driven interactions.

Other aspects of D3 concern animated transitions, interpolators and many modules for simplifying common visualization tasks (e.g., shapes and scales).

Some features of the framework required the use of other libraries like jQuery (`jquery.com`), for further DOM manipulation and asynchronous retrieving of data, Moment.js (`momentjs.com`), for efficient date and time management, and JSONPath (`code.google.com/p/jsonpath`), for efficient data extraction from JSON structures.

4.1.1 Server Side

The visualizations do not take in input the raw sensors logs. The input data is a JSON structure following some rules in order to grant a correctly rendered output. Thus, the joining link between the raw data and their visual representation is the JSON structure.

This work implemented a server-side application which creates and responses the specific JSON object based on the sensors log files. This server-side application has been deployed on Google App Engine (`appengine.google.com`), a platform for developing and hosting web applications in Google-managed data centers. This choice has been done for two reasons: (i) providing a remote access to the input data, avoiding to copy all the log files on each development machine; (ii) the simplicity and widespread use of this platform. This choice has only illustration purposes. The framework can work with input data structure contained in variables, local files, or wherever the user prefers.

The architecture used in this work (see Figure 4.2) is composed of a series of Java Servlets which parse the sensors log files, and compute the required statistics. The client application requires the input data with asynchronous requests to the server-side application which will return a JSON object. After that, the JSON object is passed as argument to the specific visualization and the result is displayed. The choice of JSON format is due to its popularity and its strong integration with JavaScript, making it less verbose in reading and writing operations, and well performing.

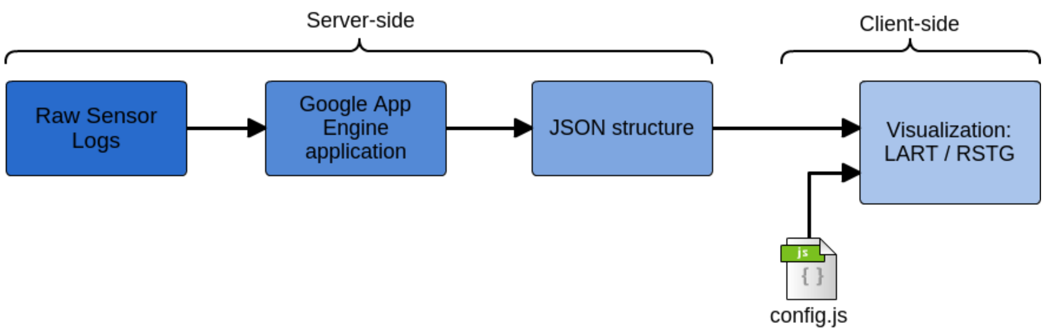


Figure 4.2 Architecture of the application.

The framework is designed to work with data collected from an AAL system. This kind of data is the product of sensors activity and it is strongly time-related. In this way, data can be sorted by the time the event occurred, creating a sequence of ordered activities. The abstraction level of this framework is raised up by the type of data it can handle: percentage, ordinal, numerical and categorical values can be reproduced in the offered visualizations. Each specific visualization requires a specific data structure, formatted in a specific way, and with some specific constraints to follow. Each specific structure will be detailed in the next sections.

4.2 Implemented Visualizations

The framework, following the theoretical principles described in Chapter 3, proposes two types of visualizations: (i) Layered Aggregate Radial Tree and (ii) Rich State Transition Graph. The former is oriented to a macro representation of the data, emphasizing the general trends, while the latter is a micro data representation, getting a deeper and complete description of a single element. Both the visualizations are surrounded by additional elements, activated through user interactions. These elements have a central role in the perception and sense-making target of this work.

In the following sections, the whole set of features and available operations of both the visualizations are described. In particular, the visual structure and how data are visually coded are explained, followed by an analysis of the input JSON structure. Finally, the surrounding interactive modules are listed, explaining their characteristics and the tasks they perform. LART, being a variation of sunburst diagram, requires hierarchical data. The information related to each node of the hierarchy is represented through colors. RSTG instead, is a variation of the well-known node-link diagram. Colors, nodes size, and arcs position are the elements that permit RSTG to represent the underlying information. RSTG input data are structured in two JSON objects: the first contains information about nodes representation, while second makes it possible to define the arcs extremes and position. The interaction modules placed alongside the visualizations permit to increase the exploration and analysis capabilities of the framework. In RSTG, filtering task is performed by the Timeline Diagram tool. In LART this task is made available by Selection tool that enables data comparison too. LART's mouse interactions make it possible to perform zooming operations on a subset of the whole dataset. Finally, the LART's Similarity tool permits the user to discover similar elements in the dataset.

4.2.1 Layered Aggregate Radial Tree - LART

LART is an interactive compact representation of hierarchical data at various aggregation levels that facilitates pattern discovery and comparison. LART shows an overview of the dataset. It is possible to point the visualization only on relevant properties subset avoiding data overload. LART is inspired by sunburst diagram. The key aspect of this kind of diagram is the hierarchical relationship of the data. For example, sunburst diagram is used in Linux file system to represent the disk usage, because it allows the user to easily detect files and folders that occupy more space. The enhancement of LART

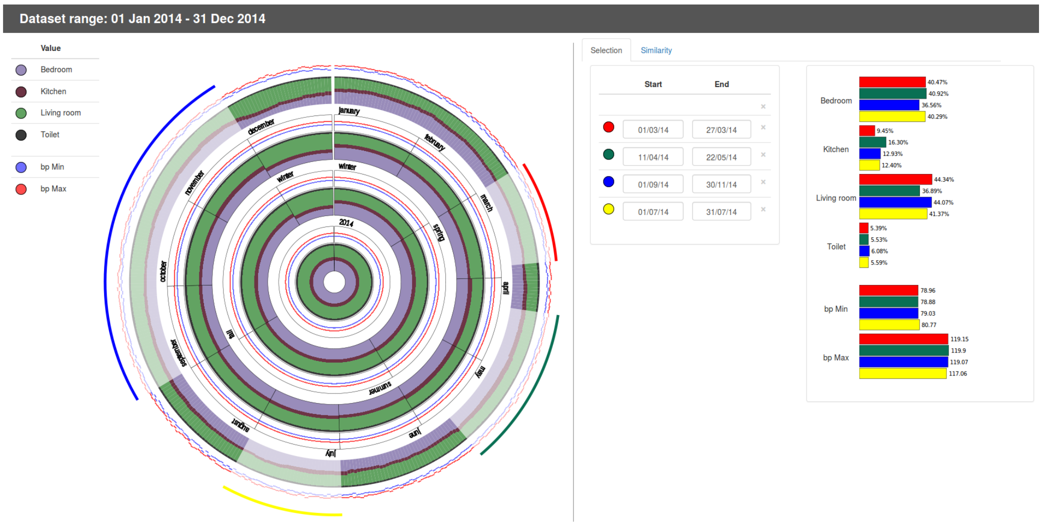


Figure 4.3 Example of LART and Selection module.

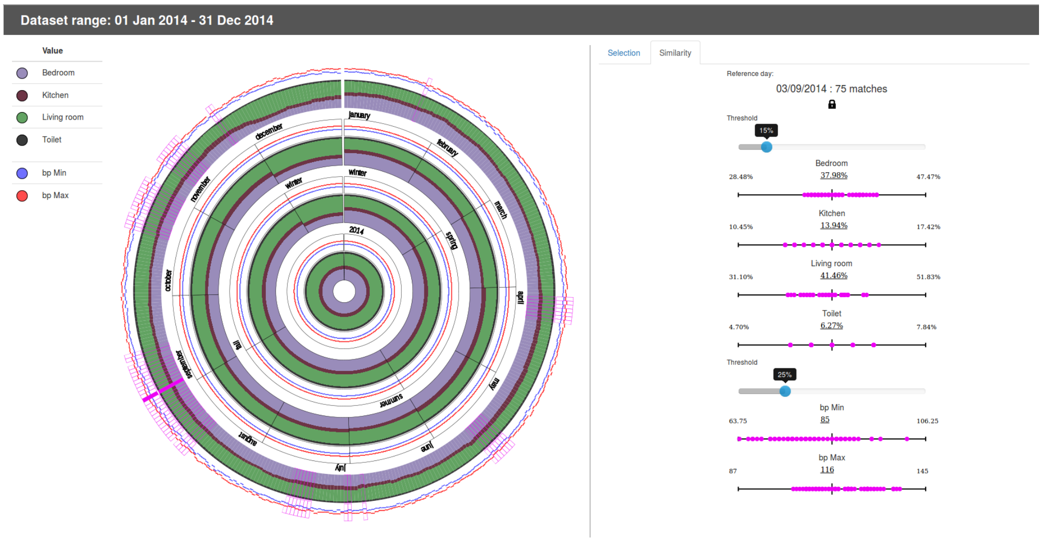


Figure 4.4 Example of LART and Similarity module.

is to represent resident's behavioral parameters alongside the hierarchical relations.

Taking into account that data collected from sensors activity are strongly time related because every sensors event has a timestamp indicating when it occurred, LART has been principally designed to work with time-related data. Exploiting the natural time hierarchies, it is possible to represent the behavior of the resident based on some particular factors (e.g., position inside the home, temperature, water usage) recorded for a long time frame (e.g., years). However LART can work with not time-related data too. To take full advantage of this representation, data must have a hierarchical relation (e.g., numbers of sensors events fired for each floor of the habitation, room, region of the room, etc.). A workaround can be done preparing a fictitious hierarchy relation: setting a figurehead root element, and using a collection a sequential data, will produce a single-layer LART which can be nevertheless helpful in the desired task. The decision-making activity is improved thanks to the surrounding interaction tools that assist the user in gaining a correct insight of the proposed problem. Figure 4.3 and Figure 4.4 illustrate a general structure of LART and all its interactive modules. Finally, it is possible to have two LARTs interlaced, one as the primary graph and the other as the secondary graph. Interlaced LARTs is useful for visually exploring the possible correlations between two data sets each represented by one LART. Figure 4.5 depicts an example of interlaced LART where the primary LART represents the amount of time spent in each room, and the secondary one represents blood pressure values.

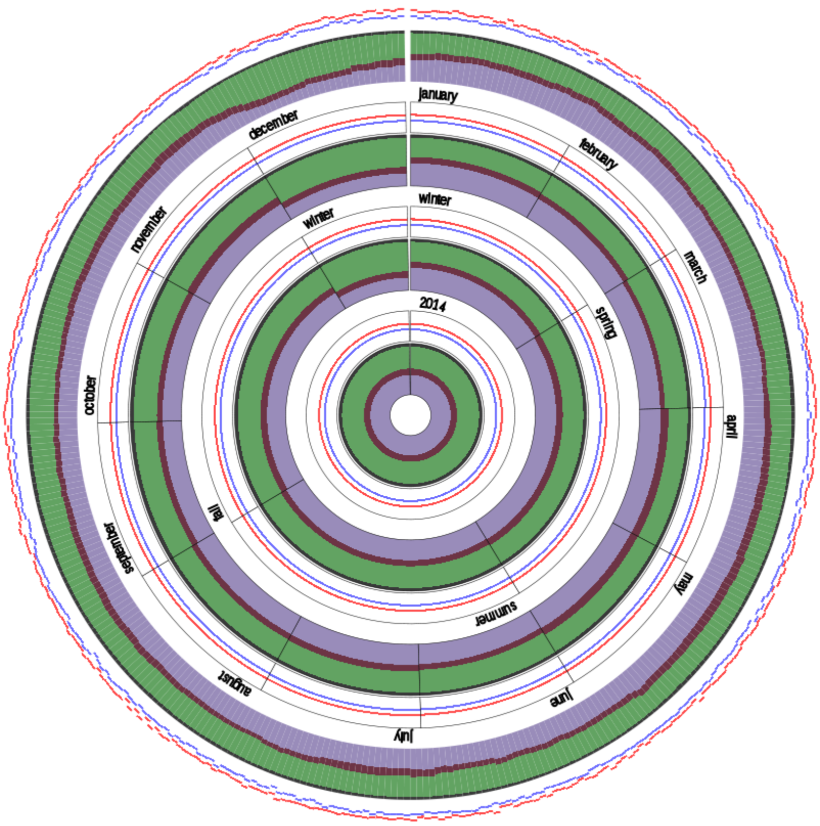


Figure 4.5 Example of LART.

Structure

LART is structurally composed of a series of concentric layers each representing an aggregation level. Aggregation levels are based on hierarchical relations: they can be time-related (e.g., years, seasons, months, days, etc.), or not (e.g., floors of the home, rooms, areas of the room, etc.). Each layer consists of data slots placed side by side with data belonging to the same hierarchy level. The innermost layer is dedicated to the root element, while the outmost layer represents leaf nodes data. Figure 4.5 shows an example of time-related interlaced LART: its hierarchical relation is years, seasons, months, days (depth = 4).

Dataset

The nature of LART requires data organized in a well defined hierarchy. Examples of hierarchies can be based on time relations (e.g., years, months, days), topological aspect (e.g., building, floors, apartments, rooms), or activity-related (e.g., kitchen-events: cooking, washing, cleaning), just to mention some examples.

The data collected by sensors are more likely to be related to the last level of the hierarchy, so in the data structure only the leaf nodes contain data regarding the residents behavior. The inner layers data are aggregated iteratively starting from the lower levels.

Technically, each node is a JSON object that, independently from the level it belongs to, must have the following properties:

- `id`: a unique identifier for that node;
- `name`: the displayed node label;

Then, there is a distinction between non-leaf nodes and leaf nodes. The former must have the property `children`, an array containing all its direct children: it acts as the connecting link from one level to the subsequent. The latter contains all the information regarding the resident behavior. These information are formatted as a series of properties named `valueXX`, where `XX` is an arbitrary sorting value used to localize a specific property. The theoretical limit of the number of properties that LART can visualize is about one hundred (from 01 to 99), but practically speaking, using more than 10 values can be counter-productive in terms of clarity and visual expressiveness (except for very particular cases). In the demo presented in Chapter 5, data have a hierarchical structure based on natural time-related hierarchy:

year, season, month and day. Each element at day-level has four properties (from `value01` to `value04`) regarding the amount of time spent in each room of the house (Bedroom, Kitchen, Living Room, and Toilet), and two properties (`value05` and `value06`) representing, respectively, diastolic and systolic blood pressure. Figure 4.7 shows an extract of the JSON input.

```
{
    "id": "level_0",
    "name": "rootElement",
    "children": [
        {
            "id": "level_1_element_0",
            "name": "First Layer Element",
            "children": [
                {
                    "id": "level_2_element_0",
                    "name": "Child #1",
                    "value01": 15,
                    "value02": 45
                },
                {
                    "id": "level_2_element_1",
                    "name": "Child #2",
                    "value01": 8,
                    "value02": 19
                }
            ]
        }
    ]
}
```

Figure 4.6 Simple example of LART input data.

```
{
    "name": 2014,                                          // Year: 2014 node (level 0)
    "id": 2014,
    "children": [
        {
            "name": "winter",                              // Season: Winter node (level 1)
            "id": "winter14",
            "children": [
                {
                    "name": "january",                     // Month: January node (level 2)
                    "id": "jan14",
                    "children": [
                        {
                            "name": "01/01/2014",          // Day: January 1, 2014 (level 3)
                            "id": 1,
                            "value01": 0.4250870943069458,     // Bedroom
                            "value02": 0.12195122241973877,    // Kitchen
                            "value03": 0.40069687366485596,    // Living Room
                            "value04": 0.04878048598766327,    // Toilet
                            "value05": 79,                     // BP min
                            "value06": 116                     // BP max
                        },
                        {
                            "name": "02/01/2014",          // Day: January 2, 2014 (level 3)
                            "id": 2,
                            "value01": 0.4285714328289032,     // Bedroom
                            "value02": 0.11846689879894257,    // Kitchen
                            "value03": 0.40069687366485596,    // Living Room
                            "value04": 0.04878048598766327,    // Toilet
                            "value05": 76,                     // BP min
                            "value06": 119                     // BP max
                        },
                        {
                            "name": "03/01/2014"
```

Figure 4.7 Example of LART input data.

During the rendering of LART, the non-leaf nodes are integrated with the aggregated values computed on top of their direct children. Figure 4.8 shows a simple example of aggregation. Starting from the lowest levels, each node is upgraded with the mean value of its direct children, which will be used for displaying the resident's behavior related to that node.

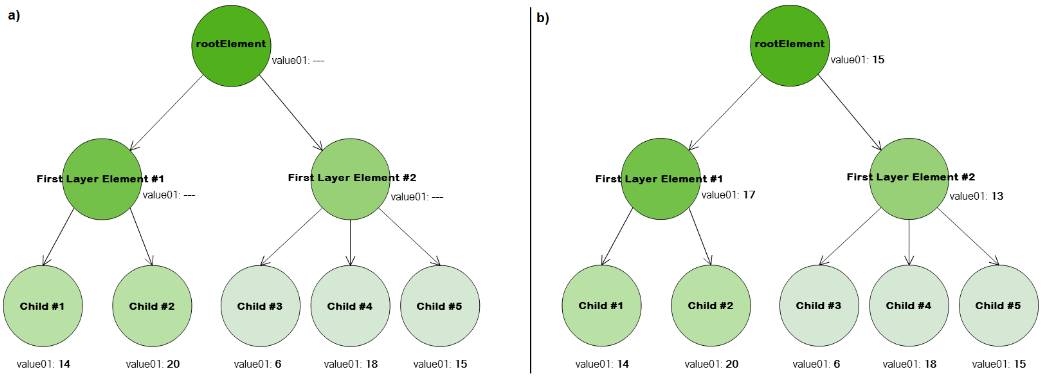


Figure 4.8 Simple example of layers aggregation: initial situation **(a)**, and after aggregation is performed **(b)**.

The unique constraint required by this structure is that leaf nodes' `id` must be sequential. This is a fundamental requirement for the Selection interactivity. However it can be managed easily. For example, it can generally be an incremental counter, or in the case time-related dataset, it could be the *day-of-year* value.

Data Coding

Data are coded based on color. They are sorted by data structures property name (i.e., first `value01`, then `value02`, `value03`, etc.) and are painted in ascending order starting from the inner circumference of the layer. Since many types of data can be represented, LART allows to paint data in two different ways: they can be rendered as areas or as lines. The former method can be useful for categorical and ordinal data, while the latter is more appropriate for numerical or absolute data. Figure 4.9 shows an example of interlaced LART where data are coded in two different ways: the primary LART represent percentages values coded with `fill` method, while the secondary LART uses `line` method for absolute values. LART takes in input also a `config.js` file which contains the parameters that permit to regulate and set up data default values. Among these parameters, the array `value.color` defines the default values colors following the painting order, while `value.fill` specifies the painting method (fill or line). The arrays `RadialTree.LAYER_SPACING` and `RadialTree.LAYER_WIDTH` define, respectively, the inner radius and the width of each singular layer. Changing these parameters allows the user to see different rendering results, and to decide which is the combination that produces a better visualization. Figure 4.10 illustrates an example of `config.js` file. In particular, lines `10` and `13` define the `fill` method and the values color, while lines `18` and `19` define the arcs inner radius and width.

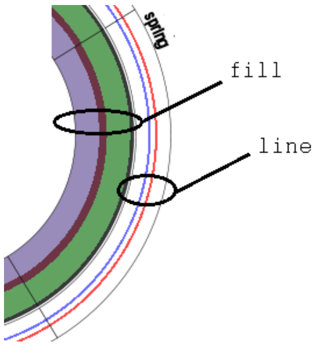


Figure 4.9 Two examples of how data are coded in LART. Inner circumference: `fill` method; outer circumference: `line` method.

```
 4var config_1 =
 5{
 6    "value": {
 7        "prefix": "value",
 8        "type": "percentage",
 9        "visualize": [1, 4],
10        "fill": "fill",
11        "id": ["bedroom", "kitchen", "livingroom", "toilet"],
12        "label": ["Bedroom", "Kitchen", "Living room", "Toilet"],
13        "color": ["RGBA(75,0,130,1)", "RGBA(178,34,34,1)", "RGBA(173,255,47,1)", "RGBA(70,130,180,1)"],
14    },
15    "RadialTree": {
16        "prefix": "RT1",
17        "count": 0,
18        "LAYER_SPACING": [20, 52, 52, 52],
19        "LAYER_WIDTH": [50, 50, 50, 50],
20        "Tween_LAYERS_SPACE": 30,
21        "Tween_LAYERS_WIDTH": 10
22    }
23};
```

Figure 4.10 Examples of LART config.js file.

Interactive Modules

User interaction is a very important component for gaining insight into the data. Basically it improves the exploratory task, giving precise indication when and where it is desired. Moreover, interaction operations allow to zoom in to a particular subset of elements and filter data focusing the analysis on one or more selected subsets. In particular, interactions provide additional tools that help to better understand the analyzed situation.

- Value Legend
  The values represented in LART are not self-explanatory. For example, it is possible to have two values regarding the minimum and maximum temperature in the home measured in a specific day, and other two values expressing the minimum and maximum residents blood pressure measured at a specific time of the day. Plotting these two sets of information produce an ambiguous result. In order to have a clearer visualization, a legend is required. The Value Legend panel lists all the rendered values. Each line of the panel refers to a value and is composed by the label assigned to that value, and its rendered color. In the `config.js` file, a JSON object is assigned to each visualized

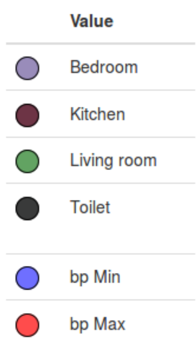


Figure 4.11 Example of Value Legend panel.

  LART. These objects contains many properties about the visual outcome. The Value Legend labels are defined in the array `value.label`, while the default color in `value.color` array. In Figure 4.10, lines `12` and `13` define the Value Legend labels and the values default colors, respectively. Colors can be manually changed by clicking on them. A color picker will permit the end-user to change the respective values color and immediately watch the rendered result.

- Tooltip
  When the pointer is over a LART's element, a tooltip pops up (see Figure 4.13). This tool shows the detailed value of the displayed resident behaviors characteristics (the `valueXX` defined in the input data structure). It helps to understand which are the reference values of the

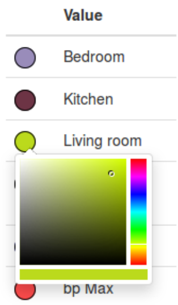


Figure 4.12 Example of Value Legend panel color change.

visualization and can be used for precisely quantify how much two, or more, elements differ one each other.

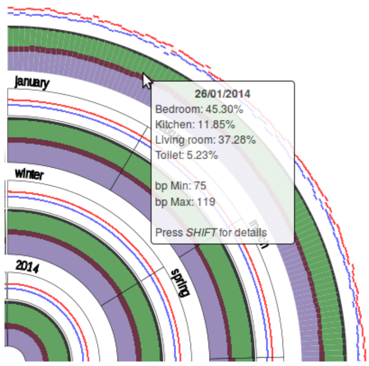


Figure 4.13 Example of LART Tooltip.

- <u>Zoom</u>
  Zooming operations make it possible to prune task-useless elements, focusing the attention only on the relevant elements subset. In presence of a very big dataset or very depth hierarchical structure (these facts may not be correlated) zooming operations are extremely useful. Zooming is performed by double-clicking a LART's element. This action produces a shrinking of the inner layers, and an expansion of the elements directly related to the selected element (see Figure 4.14). In this way, the subset including all the direct children of the selected element are highlighted. During zooming, a transition effect is used for avoiding a brutal variation of the visualization. Zoom out operation is performed by double-clicking on inner layers. It has been decided to disallow zoom actions for the outermost layer elements since it would show only the selected element. This does not contribute to any par-

ticular benefit, interfering with the rational use of the other interaction methods.

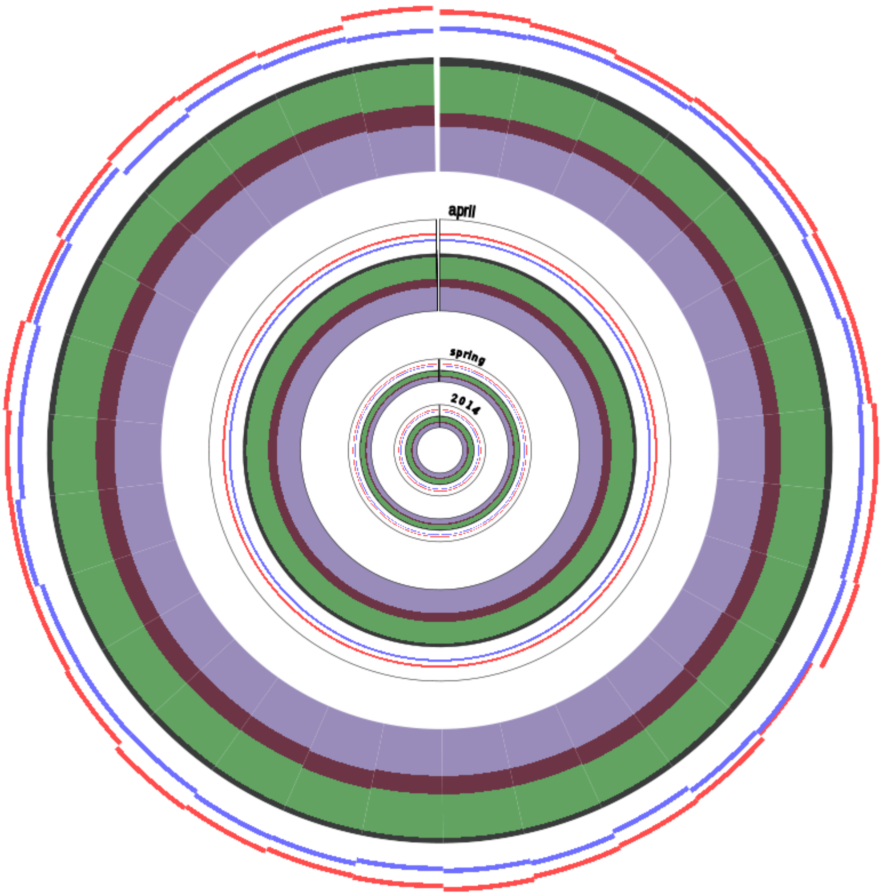


Figure 4.14 Example of zooming on April node.

- Selection

  In the context of exploration of the daily life of an older adult, it may happen that caregivers would like to compare the seniors behavior in slots of different sizes. Augmenting the dataset granularity by adding hierarchical levels may make the visualization heavier and less expressive. Relying on the caregivers eyes and memory do not lead to a reasonable solution. With the help of the Selection tool it is possible to compare the aggregated values of different groups of elements.

  A selection is a subset of the entire dataset composed by adjacent elements and defined by a start and an end element. All the elements contained between those two, belong to that selection. This is the reason why the leaf nodes of the dataset must have an ordered `id` property. Moreover, each selection is identified by an index which is used to assign it a color, improving its visual perception. Selection can be performed in two different ways:

- *drag-and-drop* on elements of the outmost layer;
- *single-click* on elements of the inner layers;

The first way creates a selection of arbitrary size, while the second method accomplishes a quick selection, selecting all the children of the reference element. Selections can overlap, both partially and completely, because it may be helpful to compare the aggregated values of selections spread on different layers (e.g., compare a month with its season). Furthermore this option increases the flexibility of this kind of interaction. When a selection is performed, the interested elements are faded, in order to highlight the selections interval. Moreover, an arc covering the selected elements is drawn. This arc does not overlap LART and it is painted with a color based on the selections index property. Selection arcs have been designed also for disambiguate overlapping selections (see Figure 4.15).

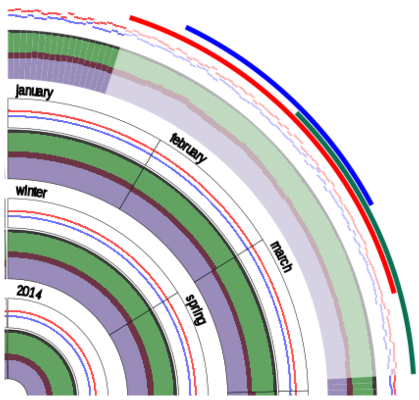


Figure 4.15 Example of overlapping selections.

If at least one selection exists, Selection Panel is displayed. It consists in a series of related bar charts. For each of the values regarding the resident's behavior represented in LART, a histogram is built, in order to compare that value's amount for each selection (see Figure 4.16b). The joining link between LART and Selection Panel is the color assigned to each selection: the bars of the histograms that share the same color

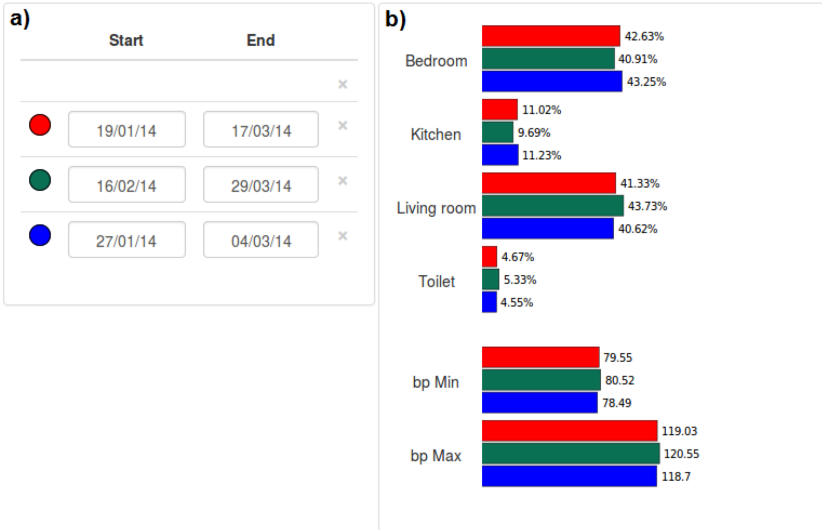


Figure 4.16 Example of Range Panel **(a)** and Selection Panel **(b)**.

represent the selection identified by the arc with that color.

Beside this area, there is a Range Panel (Figure 4.16a). It acts as the selections legend: for each selection its assigned color and its interval limits are displayed. In case of time-related data, it is possible to manually change both the selection intervals. This is helpful in presence of dense datasets, where the width of the outmost layer elements may be not so comfortable to allow a careful selection. In the Range Panel it is also possible to remove a single selection, or the whole selections set. This action provokes the removal of the corresponding arc and histogram bars.

When zooming is performed after a selection activity, both the Range Panel and Selection Panel undergo a change. Every time a zoom action is done, it defines a new visual interval: in the initial situation this interval coincides with the dataset interval, but zoom-in and zoom-out operations make it change to the zoomed element bounds. Selections that intersect the updated visual interval are temporarily split: for each part, the corresponding statistics are computed and then visualized in the Selection Panel. Only the entirely contained selections (and selection parts) are drawn in bright color. The remaining ones are disabled and they are faded both in Range Panel and Selection Panel. In such a situation, delete is allowed only for "non-split" selection in order to avoid misleading actions. Zoom-out will restore the starting situation

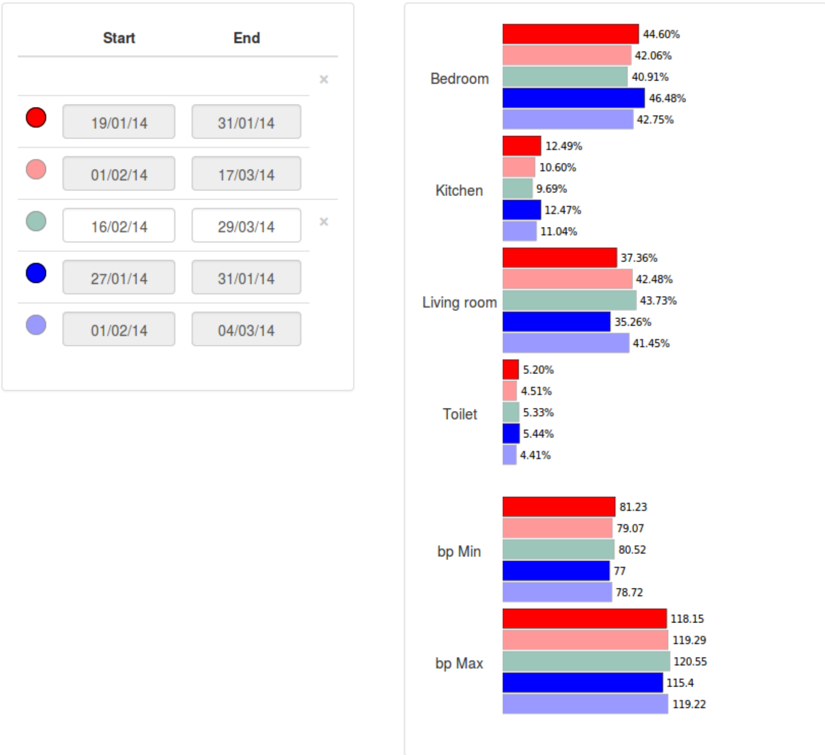


Figure 4.17 Example of Zoom on January 2014 alongside the previous Selections.

for the split selections.

- Similarity
  Similarity permits the user to understand which elements are similar. By pointing the cursor over an element of the outmost layer, all the other elements that are considered "similar" get highlighted. The metric used to define the similarity is based on the represented resident's behavioral parameters: only the elements which have all the respective values inside the interval

$$[\texttt{reference.valueXX - thr} \ ; \ \texttt{reference.valueXX} + \texttt{thr}]$$

will be considered acceptable. The pointed element is considered as `reference`. The amplitude of the interval is defined by a threshold `thr`, expressed in percentage, that can be specified through a slider. Figure 4.18 show a simple example of this mechanism, where the reference day is set to February 10, 2014, and the threshold is set to 50%. Since the reference value for 'Bedroom' is 43.21%, the interval of acceptability is

[43.21 - (0.5 * 43.21) ; 43.21 + (0.5 * 43.21)] = [**21.60%** ; **64.81%**]

The same operation is done with all the other values, and only the days that satisfy all the conditions and considered “similar”.

The side Similarity Panel provides more information regarding the distribution of data in similar days. It shows the count of matches for the current reference day and for the specified threshold. Moreover, for each represented value a line graph is plotted, expressing the kind of distribution of the similar days for that specific value. For each point belonging to a line graph it shows the precise value, and a single-click will pop out the possible similar days which have in common the same value for that parameter. In presence of interlaced LART, two thresholds must be specified, taking into account the correlation between the two different groups of information.

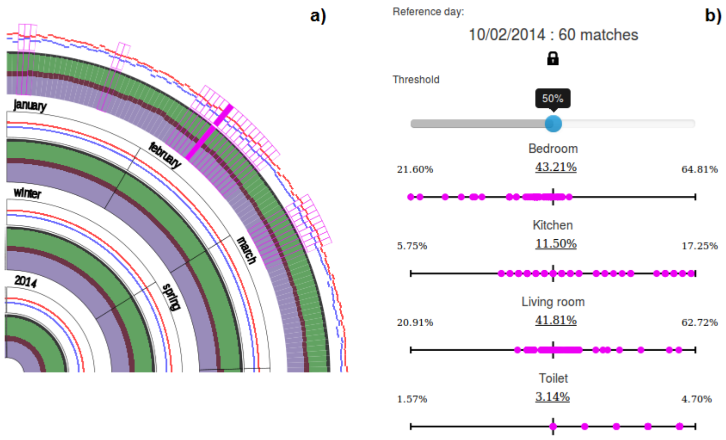


Figure 4.18 Example of Similarity **(a)** and Similarity Panel setting February 10, 2014 as reference day, and a similarity threshold of 50% **(b)**).

4.2.2 Rich State Transition Graph - RSTG

RSTG is the representation of a single group of information, visualizing state transitions of sequential temporal data keeping time, duration, and frequency information. It is a variant of node-link diagram, a kind of visualization useful to show the transitions from one state to another. RSTG makes it possible to get details about resident’s behavior, giving complementary information to the ones offered by LART.

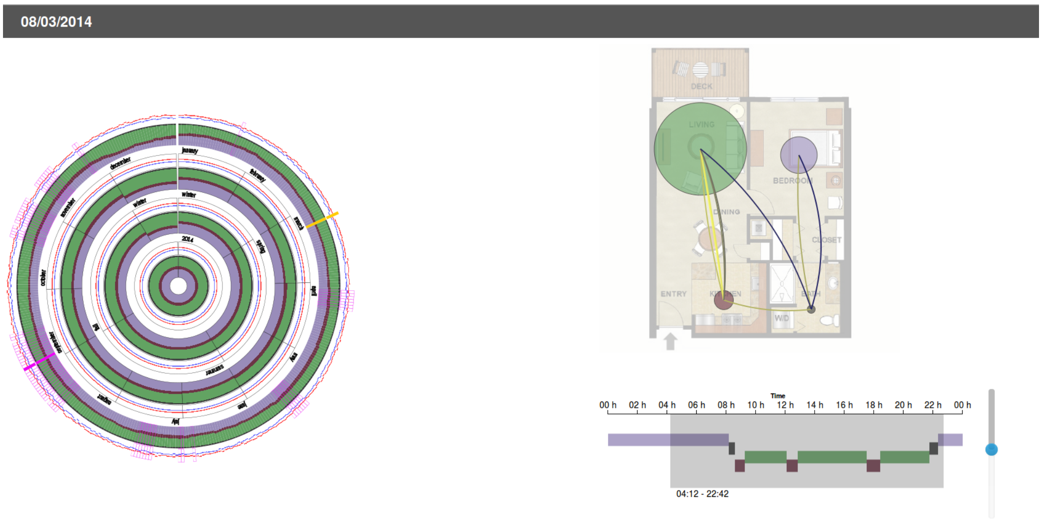


Figure 4.19 Example of RSTG with Timeline Diagram.

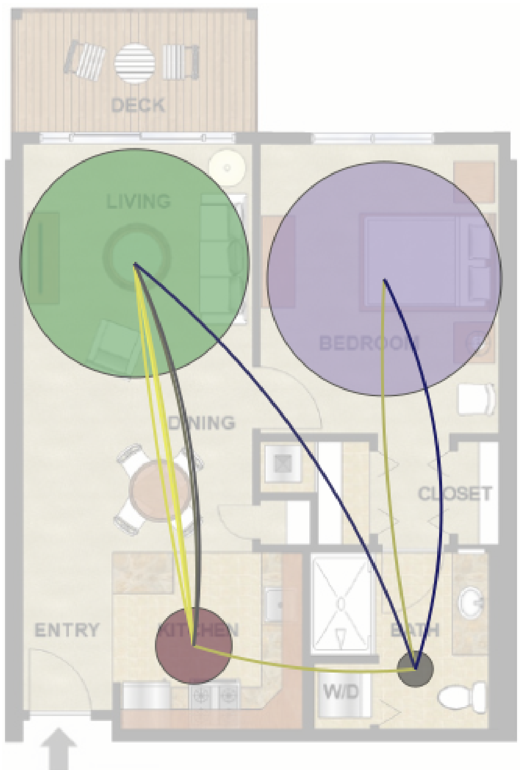


Figure 4.20 Example of RSTG.

Structure

RSTG is structurally very similar to a classic node-link diagram: it is composed of nodes represented by circles and links represented by arcs connecting one node to another one. Each node represents a state defined on top of the underlying dataset. Each link represents a transition from one state to another state. Multiple arcs are allowed and are used to show the amount of transitions between two states. Figure 4.20 shows an example of RSTG visualization.

Dataset

Data for RSTG should be categorical (or ordinal and finite). It does not mean that RSTG cannot be used for numerical data but only that numerical data need to be discretized before. For example, blood pressure can be discretized to low, normal, and high: each category is represented by a node of radius proportional to the frequency of occurrence, and arcs represent the trend of blood pressure in the selected time slot.

Two JSON structures are used in this visualization. The first contains information about the nodes, precisely the quantity that will define the nodes radius. Figure 4.21 shows the first input data structure taken from the demo of Chapter 5. For each room, is defined the "percentage" value which will be used in order to compute the node's size. The second structure is an array listing the initial and ending time slot for each activity. This one is shared between both RSTG and Timeline Diagram: the former uses it for deriving the transitions to be drawn (start&end nodes, left-to-right position, and color), the latter uses it to create each activity block and plot the timeline. Figure 4.22 illustrates an extract of the second JSON input, where is listed the sequence of resident's position in the house on a given day. Each group of consecutive locations in the same room is associated with its initial and final time slot's IDs (in this example, a time slot is five minutes long).

Data Coding

The variations introduced to general node-link diagrams carry the data information: nodes have no equal size, and more links connect one node to another. Nodes are colored following the values color code specified in LART visualization, and their size is proportional to the total amount of time spent in a state for the time period of interest. Links connecting the nodes are color coded and arranged in a left-to-right/bottom-to-top manner to indicate the timing of the transitions: the more a link is on the left, the lighter its color is, representing a transition occurred in early hours of the day, and vice versa for links shifted on the right.

```
{
    "nodes": [
        {
            "type": "B",                          // Bedroom
            "count": 116,
            "percentage": 0.40418118238449097
        },
        {
            "type": "K",                          // Kitchen
            "count": 28,
            "percentage": 0.0975609719753265
        },
        {
            "type": "L",                          // Living Room
            "count": 130,
            "percentage": 0.45296168327331543
        },
        {
            "type": "T",                          // Toilet
            "count": 12,
            "percentage": 0.04181184619665146
        }
    ]
}
```

Figure 4.21 Example of RSTG first input data.

```
[
    {
        "value": "B",        // In Bedroom for 98 time slots
        "startId": "1",      // Interval start ID
        "endId": "98"        // Interval end ID
    },
    {
        "value": "T",        // Then, in Toilet for 5 time slots
        "startId": "99",
        "endId": "103"
    },
    {
        "value": "K",        // Then, in Kitchen for 8 time slots
        "startId": "104",
        "endId": "111"
    },
    {
        "value": "L",        // Then, in Living Room for 34 time slots
        "startId": "112",
        "endId": "145"
    },
    {
        "value": "K",        // Then, in Kitchen for 9 time slots
        "startId": "146",
        "endId": "154"
    },
    {
        "value": "L",        // Then, in Living Room for 56 time slots
```

Figure 4.22 Example of RSTG second input data.

Interactive Modules

The number of possible interactions in RSTG is lower with respect to LART since it is a simpler visualization, very self-explanatory.

- Pop-up Label
  When the pointer is over a node or a link, a label pops up explicating the underlined information. Nodes will show the total amount of time spent in that state, while links express when the transition occurred. This kind of interaction makes it possible to understand precisely which are the bounds in a dense group of close transitions. Figure 4.23 shows an example of pop-out label on a transition link: the time at which the transition happened is displayed.

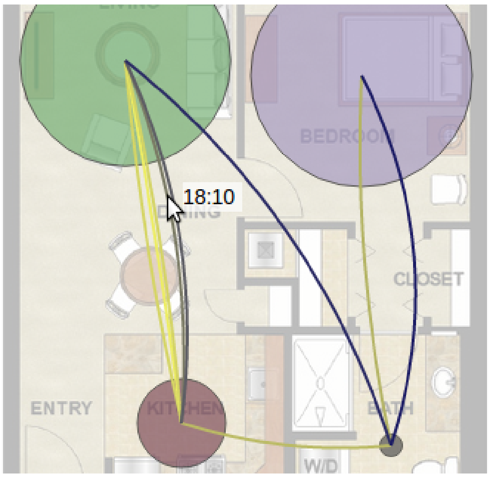


Figure 4.23 RSTG Pop-up Label on a transition link.

- Timeline Diagram
  Timeline Diagram is a complementary tool for RSTG (see Figure 4.24a). It is a timeline composed of the behavioral values, represented as blocks and painted with the usual color code. Each block represents a series of identical activities occurred in sequence. It is another way to see the situation. A slider placed next to the timeline makes it possible to separate the blocks in the vertical direction, producing a clearer picture in presence of very short activities.

  An interaction that was thought be useful is the filtering proposed by Timeline Diagram. It defines a temporal range for the visualization. By default it coincides with the entire time frame analyzed (e.g., a day of 24 hours). Performing a drag-and-drop action on the Timeline Diagram will involve the variation of the temporal range, and the recalculation of the aggregated information on top of data filtered and confined to the specified range.

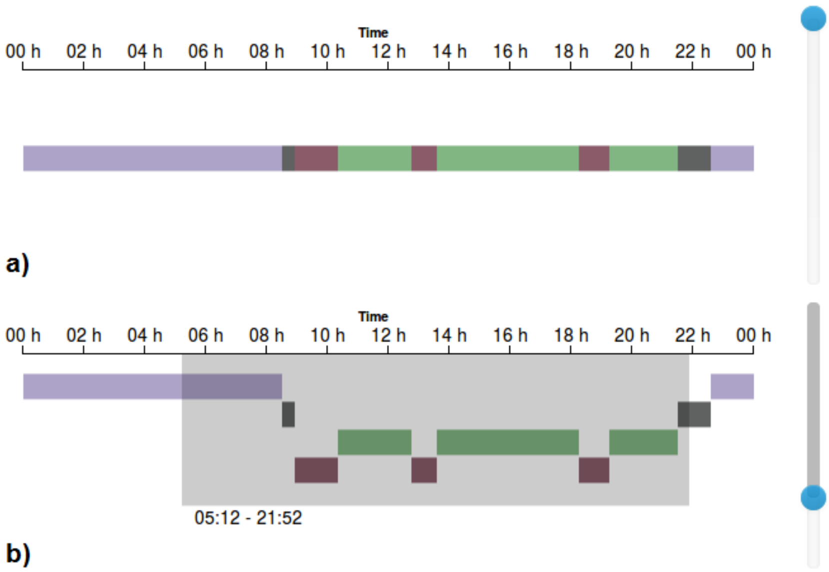


Figure 4.24 RSTG Timeline Diagram **(a)** and Range selection **(b)**.

Chapter 5

Usability Evaluation

This chapter reports an evaluation study in order to collect feedbacks and opinions about the implemented framework. For this purpose, an interactive demo has been implemented and a group of volunteer participants were asked to perform some tasks and answer to a questionnaire, entirely reported in Appendix A, for evaluating the visualizations.

5.1 Objectives of the Evaluation

The evaluation of the visualizations usability aims to understand if the developed visual representations fulfill the task of information-seeking and facilitate the exploration of large datasets. The collection of feedbacks and comments from different users makes it possible to discover possible weak points, indicating future works directions targeted to the improvement of the visualizations.

5.2 Design of the Evaluation Methodology

After a brief explanation of the use case, the participant can start to experiment the visualizations. A short tutorial describes which are the features and the most important functionalities of both the visualizations. When the participant feels confident with the application, the test session starts. The user is asked to perform five tasks and to fill a questionnaire. Each task requires to perform some operations and to provide an answer in order to understand if the correct actions has been done. Participant has to fill a questionnaire for an evaluation of each performed task. The task-specific questions, ask for:

- Task Difficulty: the perceived difficulty of the performed task;
- Visual Difficulty: the perceived difficulty of performing the task only by means of visual searching.

Once all the tasks are performed, the participant is asked to provide an overall evaluation about the effectiveness of the visualizations, of the interaction mechanisms, and about the usefulness of such a tool for a care professional. Finally, the participant is asked to leave some comments about his/her application experience. The analysis of the filled questionnaires allows a qualitative evaluation of the visualizations, useful to understand the participants' feelings about them.

During the evaluation process a user-tracking mechanism traces the participant's actions and the provided answers to the tasks. This mechanism makes it possible to collect information about the time spent by the user to perform each task, the operation performed to accomplish each task, and the correctness of his/her answers. These events are stored in log files, and their analysis permits to have quantitative results that can be studied and compared. Figure 5.1 shows an excerpt of log file: each line of the log contains the timestamp and can represent a tracked event, or the answer the user gave to a task. Possible types of tracked events are basically mouse-over and mouse-out occurrences, and the mouse position, which is traced every 200ms.

5.3 Evaluation Session

The evaluation session took place between March 2, 2015 and March 10, 2015. It involved twenty volunteer participants belonging to the academic

```
1425911118652, overview, nickname, fabiov                 // Participant nickname
1425911128392, overview, taskstart, 1                     // Task 1 started
              ...
1425911139244, overview, taskend, 1                       // Task 1 completed
1425911146901, overview, taskstart, 2                     // Task 2 started
1425911147844, overview, mouse, 486, 330                  // Mouse move event
1425911148045, overview, mouse, 485, 330                  // Mouse move event
1425911148097, overview, mouseover, 474, 330, november    // Mouse over November node
1425911148140, overview, mouseout, 441, 321, november     // Mouse out November node
1425911159244, overview, mouse, 1644, 892
1425911159417, overview, taskanswer, November             // Task 2 answer
1425911159417, overview, taskend, 2                       // Task 2 completed
              ...
```

Figure 5.1 Example of evaluation log file.

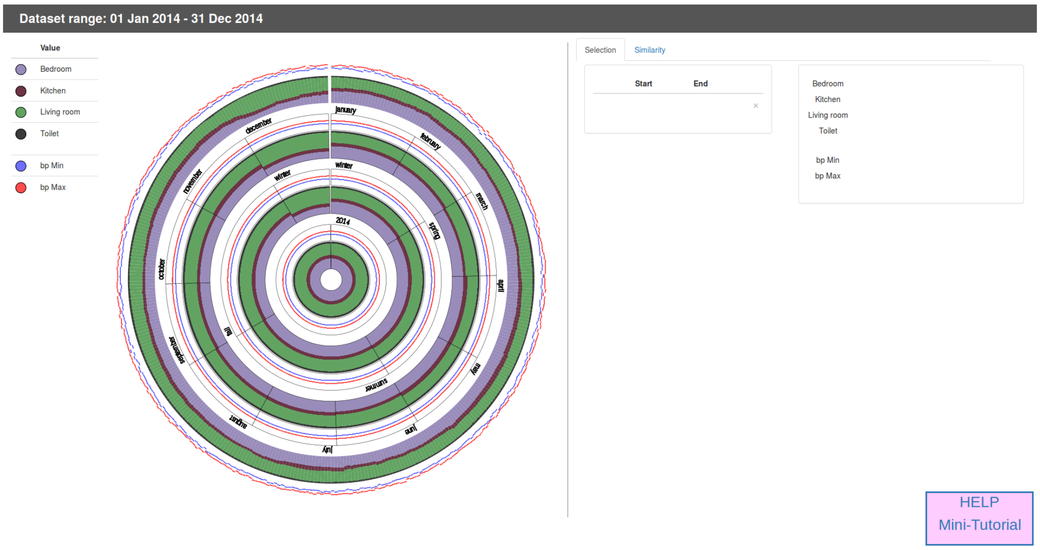


Figure 5.2 Evaluation Demo home screen.

environment. For the majority of the participants this was the first experience with the proposed visualizations, while a couple of them had already seen both LART and RSTG in the early development phase.

The use case defined for this evaluation session is the following: in the home of an aged person, a PIR sensor is installed in each room. The resident's apartment is composed of four rooms: bedroom, kitchen, living room and toilet. Moreover, the senior's blood pressure (systolic and diastolic) is supposed to be measured every day and to be collected in the system.

The implemented demo followed the classical visualization path described in the previous chapter: the home screen displays an interlaced LART visualization with all the interactive modules (see Figure 5.2), and the detailed information are displayed with the help of a RSTG visualization. The data collected in the elderly's home are:

- position of the senior in his habitation, in terms of rooms;
- systolic (max) and diastolic (min) blood pressure values;

The "position" of the resident is supposed to be sampled every five minutes, collecting the PIR sensors state. In this way, for each monitored day there are 288 samples. In this scenario, we do not take into account out-of-home events, but for the purposes of the demo this is not a lack of generalization. The blood pressure values, instead, are absolute value types, expressed in `mmHg` unity of measure. The blood pressure is supposed to be measured daily at the same time. The dataset, that has been created only for this evaluation task, represents data collected from January 1, 2014 to December 31, 2014.

Both LART and RSTG visualizations follow the directives stated in Chapter 4. LART's input data has a hierarchical structure based on natural time-related hierarchy: year, season, month and day. Each element at day-level has four properties (from `value01` to `value04`) regarding the amount of time spent in each specific room, and two properties (`value05` and `value06`) representing, respectively, diastolic and systolic blood pressure. The values regarding the rooms are expressed in percentage values, easily derivable from the sensors logs. Figure 5.3 depicts and an example of day definition. In RSTG each node is associated to a room. The node radius represents the amount of time spent in that room. The links connecting two nodes represent the transfers between the two rooms. Both the nodes and links properties are computed based on the selected time range of visualization.

```
},
{
    "name": "14/07/2014",                // Day: July 14, 2014
    "id": 195,
    "value01": 0.37979093194007874,      // Bedroom: 37.98%
    "value02": 0.1428571492433548,       // Kitchen: 14.30%
    "value03": 0.4111498296260834,       // Living Room: 41.11%
    "value04": 0.06271777302026749,      // Toilet: 6.27%
    "value05": 75,                       // BP min: 75 mmHg
    "value06": 116                       // BP max: 116 mmHg
},
{
```

Figure 5.3 Example of LART input data at day-level.

The participants were asked to perform these tasks:

Task 1 In the overview, choose the room colors in a way so that you can easily evaluate the kitchen percentage data (the colors that suit your eyes the most).

Task 2 Find the MONTH with the highest average KITCHEN percentage.

Task 3 Choose the DAY in the month you found in the previous task (`Task 2`) that has the MAXIMUM KITCHEN time percentage.

Task 4 How many SIMILAR DAYS to the day you found in the previous task (`Task 3`) with a threshold of 30% and NOT considering the blood pressure (considering ONLY the same month)?

Task 5 Get to the DETAILED VIEW of the day you found in `Task 3`. Does the person have more transitional activities during the day or during the night? How many times does the person go from the living room to the kitchen and vice versa?

The questionnaire relative to this evaluation session in fully reported in Appendix A and it follows the structure defined in the previous section.

5.4 Summary of Results

Almost all of the participants have completed the requested tasks. Two of them had found problems due to low monitor resolution, preventing the full accomplishment of the evaluation process.

The quantitative analysis reveals that great part of the answers was correct (in particular no one made mistakes in `Task 2` and `Task 3`). `Task 4` was as a matter of fact the most difficult and some participants gave incorrect answers. In `Task 5`, 90% of the answers were correct, granting a good result. `Task 1` has been excluded from this analysis since it is strongly subjective and its purpose was to understand if the default colors chosen for the demo were meaningful and visually acceptable. The average time required for completing all the tasks was about 8 minutes, which is a good result, taking into account that for most of the participants it was the very first time they saw such visualizations. The slowest user took about 16 minutes to complete the process, a value that is close to the expected time. The quickest user, which had a previous short experience with LART and RSTG visualizations, used only 2.5 minutes to correctly answer to all the questions: this reveals that even with a modest experience it is possible to gain correct results in short time. Figure 5.4 summarizes the evaluation results taking into account the correctness of the answers and the needed time to perform the tasks. Moreover, it excluded the users who had trouble in the evaluation accomplishment or have not entirely completed the evaluation process.

The qualitative results extracted from the filled questionnaires are illustrated in Figure 5.5. As expected, `Task 4` revealed to be the most difficult to accomplish since it involved many interaction tools and many conditions for reaching the right result. Despite that, its Visual Difficulty value shows that the involved interactive module (in this case, Similarity Panel) clearly represents the underlying information. Visual Difficulty and Task Difficulty values of `Task 2` and `Task 3` are in the norm, since they are two simple tasks and

	T1	T2		T3		T4		T5		
	time [sec]	correctness	time [sec]	correctness	time [sec]	correctness	time [sec]	correctness	time [sec]	total user time (wo T1) [sec]
u1	10.852	y	12.516	y	32.696	y	22.221	y	89.358	156.791
u2	57.203	y	72.74	y	83.132	y	57.579	y	234.344	447.795
u3	44.994	y	71.187	y	134.171	n	127.168	y	269.66	602.186
u4	22.86	y	49.169	y	246.421	n	145.032	y	187.173	627.795
u5	18.24	y	47.545	y	133.489	n	145.997	y	113.117	440.148
u6	95.993	y	56.124	y	114.258	n	135.466	n	190.434	496.282
u8	170.429	y	183.781	y	241.947	y	195.861	y	333.65	955.239
u9	36.108	y	74.913	y	104.926	n	71.985	y	126.463	378.287
u10	31.078	y	134.316	y	287.064	n	91.614	y	164.307	677.301
u12	11.968	y	78.45	y	164.822	y	65.074	y	58.426	366.772
u13	91.036	y	125.709	y	129.281	y	98.956	y	59.883	413.829
u14	18.835	y	96.993	y	51.911	y	131.588	y	196.319	476.811
u15	15.5	y	48.344	y	88.331	n	129.117	y	123.894	389.686
avg	48.08430769	93.80%	80.90669231	87.50%	139.4191538	43.80%	109.0506154	75.00%	165.156	**494.5324615**
std	46.33977893	45.0711489		77.03882384		46.57860075		81.81313311		**191.8925654**
min	10.852	12.516		32.696		22.221		58.426		**156.791**
max	170.429	183.781		287.064		195.861		333.65		**955.239**

Figure 5.4 Results of the quantitative evaluation.

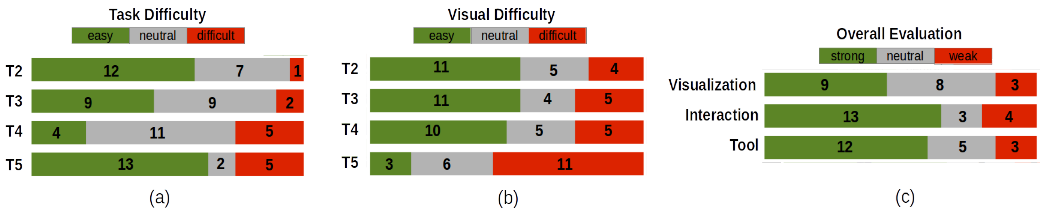


Figure 5.5 Results of the qualitative evaluation.

the 80% of the participants had no difficulties in gathering the relevant information. Surprisingly, the high value of Visual Difficulty for `Task 5` indicates a general complexity in retrieving the information for the RSTG visualization, although the Task Difficulty value indicates that it is easy. This aspect requires further investigations in order to understand which are the reasons of such an low evaluation.

Most of the participants believed that the proposed visualizations are a good tool to accomplish the monitoring of the elderly's daily behavior. In particular, the interaction tools have been considered helpful for reaching that kind of result. The participants' comments confirm that the overall application is considered positive. In particular, they appreciated the fact that it simplifies the visualization of data structured in hierarchical way, making possible to handle a great amount of data. A participant claimed that despite the fact that at a first glance the visualizations (in particular LART) seem a bit complicated, in short time it gets really easy to work with it. It has been appreciated also the flexibility to extract data and the short time needed to perform the required tasks. Some additional features was suggested, like adding the possibility to perform a quick Selection by clicking on the year/season/month label, and the highlighting of a value when onmouse-over event occurs. The non-presence of labels at day-level has been felt as a lack of notation by one user, but this has been an implementation choice, since it would make the overall visualization heavier. A halfway may be adding labels at regular intervals.

5.5 Discussion

The analysis of the evaluation results express that the effectiveness of LART is appreciated. It can be further improved by adding new elements like, as suggested, identificative labels on the outermost layer. The evaluation session reveals an unexpected lack of expressiveness in RSTG. This aspect needs extra investigations and must be accurately analyzed in order to discover the RSTG components that have produced such a negative result. Since `Task 5` concerns transitions, it is possible that some features of the connection links may have produced a confusing result. Aspects like the links' position, their angle of curvature, their colors, or their thickness must be reviewed. Another way that could increase the effectiveness of RTSG is to introduce a series of inner circles inside each node representing continuous partial stay in that state (see Figure 5.6).

Figure 5.6 An example of possible RTSG expressiveness improvement.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This work of thesis concerned the design and implementation of a visualization framework aimed to explore and analyze data regarding the daily life of aged people, collected in an AAL system. Since the amount of data gathered from such a system is huge, data visualization tools are necessary to extract relevant information. The main target of the proposed framework is to provide a series of visualizations that can stimulate the human visual perception, in order to get an insight of the represented data and make it possible a sense-making of them. Starting from the review of the related works in the field of data visualization and visualization in AAL systems, this work illustrates the human visual perception benefits for pattern recognition tasks and how it can be exploited in data representation. The developed visualizations make it possible to have a clear understanding about some aspects of the resident's life style. Layered Aggregate Radial Tree (LART) summarizes these aspects, producing a visual data representation that emphasizes the exploration and analysis of a great amount of data. This way of presenting information gives the opportunity to have a picture of the resident's behavior for a long time period, making it possible to understand if particular variations in the daily habits may require further investigations. Rich State Transition Graph (RSTG) displays information at a lower granularity level, proposing statistics that cannot be shown in LART for clarity and space reasons. RSTG integrates the information discovered by LART giving details on demand. Both the visualizations are accompanied by interactive modules that foster the dataset exploration and the information extraction.

6.2 Future Work

This work marks the starting point of a series of activities concerning data visualization in the ATG group. So, there is still a long way to reach a mature solution. There are mainly three directions of future works in order to improve the capabilities of the proposed work. The first path concerns the internal structure of the visualizations and their rendered outcome. Then, back-end improvements may be done in order to assure better performances. Finally, an automatic mechanism for the analysis of the log files containing the users' visual actions collected during the evaluation session could give further details about the visualizations usability.

Visualization

Both LART and RSTG visualizations can be improved under many aspects. First of all, the RSTG structure could be generalized and parametrized in order to get a level of flexibilty similar to LART. The parametrization work could include the definition of a configuration file similar to the one used in LART. Also some interactive modules should be updated in order to get these module available for a broader set of data. For example, the LART's Range Panel works only with time related data, excluding the opportunity to use it in other contexts. Finally, the visualizations and all the interactive modules need to be converted to a full responsive interface, making them completely available for a wider range of devices, like low resolution devices. In order to get clearer ideas on the parts that need a prior improvement, this work path must be supported by a further and maybe slightly different evaluation process.

Performance

The computational balance between server-side and client-side application can be brought into question. Interactions and exploration in real time that avoid lags and time delay offer a better user experience. Some issues in this aspect may be find in presence of very huge dataset. Optimizing the back-end performances could be the first step. Data collected by the AAL system could be arranged in databases, improving the statistics computation performances. In the visualization process, caching methods can be introduced in order to improve the response time of particular interactions (i.e., LART Zooming). In this direction, the input data structures of both LART an RSTG can be slightly changed making possible an efficient use of shared visualization elements (e.g., SVG <radialGradient> elements).

Evaluation

An automated evaluation log analysis mechanism should be taken into account. The information contained in the log files collected during the evaluation process can be better analyzed. A software could parse and analyze a log file and give accurately information about the user's actions in performing a task. Having at disposal a clear tracking of the user's experience makes it possible to get a better idea about where and which the strength and weak points are, giving even more weight to the evaluation process.

Appendix A

Chapter 5 describes an evaluation process for the visualizations proposed in this work. The participants were asked to fill an online questionnaire in order to collect opinions and feedbacks about the tasks they had to perform. This Appendix reports the entire questionnaire.

Evaluation of BridgeViz Visualizations

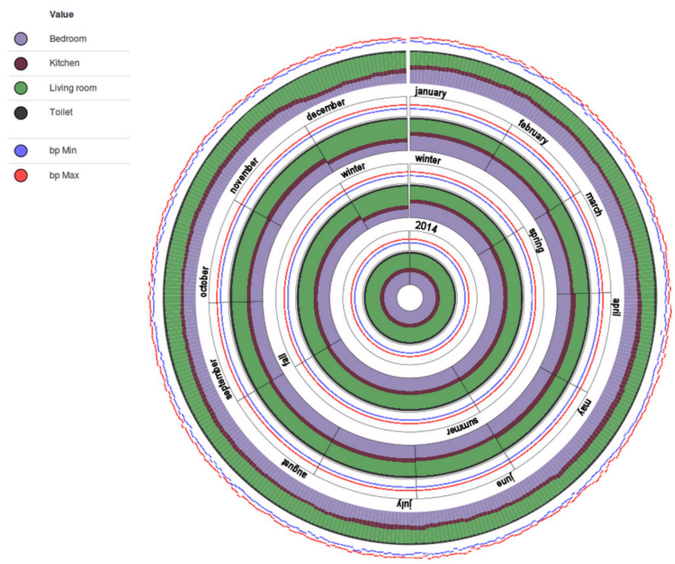
Bridge is a project for mutual reassurance of fragile people (living independently) and their family. It consists of transforming the living space of the person into a smart environment equipped with sensors, actuators, and intelligence. The collected data is stored in a database for later long-term analysis. BridgeViz is one of the mechanisms to make sense of these data through interactive visualizations. In this questionnaire we would like to ask you to kindly help us in evaluating some of our visualizations.

- This evaluation will require about 20 mins of your time.
- You are asked to perform some tasks after a mini-tutorial for how to use the tool.
- Please decide for a nickname that you need to enter both in this form and on the online website.
- Some short answers concerning the tasks you perform will be collected on the online website.
- Please fill out this form after performing the tasks (try to remember the tasks as the questions in this form mainly regard the tasks).

Thank you for your time & patience! :)

Radial Tree (Overview Visualization)

Radial Tree is an interactive impact representation of temporal data at various aggregation levels (based on natural time hierarchies) that facilitates pattern discovery and comparison. (see the figure below)



State Transition Graph (Detail Visualization)

State Transition Graph is a node link diagram that visualizes state transitions of sequential temporal data. (see the figure below)

Getting familiar with the tool

Please take some time (~ 5 min) to get familiar with the tool environment and come back to this form.
(Use FIREFOX to see correctly the visualization elements.)

Here is the link to a mini-tutorial on how to use the tool:
http://home.deib.polimi.it/saidinejad/VizDemo/help.html

And here you can use the tool available online:
http://home.deib.polimi.it/saidinejad/VizDemo/index.html

Tasks

Please follow this link in which you will get the necessary instructions to perform some tasks with the tool.
You will answer to some questions in the tool. Afterwards, in the rest of this questionnaire we will ask for your feedback on the tasks you did.

http://home.deib.polimi.it/saidinejad/Evaluation/index.html

Please give yourself a nickname (the same you used in the tool). *

Task 1

In the overview, choose the room colors in a way so that you can easily evaluate the kitchen percentage data (the colors that suit your eyes the most).

Did you find it useful to change the colors?

	1	2	3	4	5	
Not at all	○	○	○	○	○	Necessary

Task 2

Find the MONTH with the highest average KITCHEN percentage.

How much was it easy to VISUALLY perform this task?

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Please check the elements that you used for performing this task.

- ☐ Month Layer Tool Tip (mouse hover)
- ☐ Selection Panel
- ☐ Similarity Panel
- ☐ Zooming
- ☐ Other: ____

Please rate the difficulty of performing this task.

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Task 3

Choose the DAY in the month you found in the previous task (Task 2) that has the MAXIMUM KITCHEN time percentage.

How much was it easy to VISUALLY perform this task?

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Please check the elements that you used for performing this task.

- ☐ Tool Tip (mouse hover)
- ☐ Selection Panel
- ☐ Similarity Panel
- ☐ Zooming
- ☐ Other: ____

Please rate the difficulty of performing this task.

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Task 4

How many SIMILAR DAYS to the day you found in the previous task (Task 3) with a threshold of 30% and NOT considering the blood pressure (considering ONLY the same month)?

How much was it easy to VISUALLY perform this task?

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Please check the elements that you used for performing this task.

- ☐ Layer Tool Tip (mouse hover)
- ☐ Selection Panel
- ☐ Similarity Panel
- ☐ Zooming
- ☐ Other:

Please rate the difficulty of performing this task.

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Task 5

Get to the DETAILED VIEW of the day you found in Task 3. Does the person have more transitional activities during the day or during the night? How many times does the person go from the Living room to the kitchen and vice versa?

How much was it easy to VISUALLY perform this task?

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Please check the elements that you used for performing this task.

- ☐ State Transition Graph
- ☐ Horizontal Timeline
- ☐ Other:

Please rate the difficulty of performing this task.

	1	2	3	4	5	
Very easy	○	○	○	○	○	Very difficult

Overall Evaluation

Your overall evaluation of visualizations.
(effectiveness of visual clues to perform tasks more easily)

	1	2	3	4	5	
Very weak	○	○	○	○	○	Very strong

Your overall evaluation of Interaction mechanisms.
(effectiveness of interaction mechanisms: selection, zooming, etc.)

	1	2	3	4	5	
Very weak	○	○	○	○	○	Very strong

Your overall evaluation of the effective of such a tool for a care professional.

	1	2	3	4	5	
Not helpful at all	○	○	○	○	○	Very helpful

Thank you very much. Finally we will be glad to have your comments.

Bibliography

[1] United Nations. Dept. of Economic. *World population ageing, 1950-2050*. Number 207. New York: United Nations, 2002.

[2] J.J. Thomas and K.A. Cook. A visual analytics agenda. *Computer Graphics and Applications, IEEE*, 26(1):10–13, Jan 2006.

[3] Katerina Vrotsou, Mathias Bergqvist, Matthew Cooper, and Kajsa Ellegård. Podd: A portable diary data collection system. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, AVI '14, pages 381–382, New York, NY, USA, 2014. ACM.

[4] Katerina Vrotsou, Jimmy Johansson, and Matthew Cooper. Activitree: interactive visual exploration of sequences in event-based data using graph similarity. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):945–952, 2009.

[5] Marc Weber, Marc Alexa, and Wolfgang Müller. Visualizing time-series on spirals. In *Information Visualization, IEEE Symposium on*, pages 7–7. IEEE Computer Society, 2001.

[6] Catherine Plaisant, Brett Milash, Anne Rose, Seth Widoff, and Ben Shneiderman. Lifelines: visualizing personal histories. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 221–227. ACM, 1996.

[7] Taowei David Wang, Catherine Plaisant, Alexander J Quinn, Roman Stanchak, Shawn Murphy, and Ben Shneiderman. Aligning temporal data by sentinel events: discovering patterns in electronic health records. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 457–466. ACM, 2008.

[8] Krist Wongsuphasawat, John Alexis Guerra Gómez, Catherine Plaisant, Taowei David Wang, Meirav Taieb-Maimon, and Ben Shneiderman. Lifeflow: visualizing an overview of event sequences. In *Proceedings*

of the SIGCHI Conference on Human Factors in Computing Systems, pages 1747–1756. ACM, 2011.

[9] Michael Burch, Fabian Beck, and Stephan Diehl. Timeline trees: visualizing sequences of transactions in information hierarchies. In *Proceedings of the working conference on Advanced visual interfaces*, pages 75–82. ACM, 2008.

[10] Gilles Virone, Majd Alwan, Siddharth Dalal, Steven W Kell, Beverely Turner, John A Stankovic, and Robin Felder. Behavioral patterns of older adults in assisted living. *Information Technology in Biomedicine, IEEE Transactions on*, 12(3):387–398, 2008.

[11] Maurice Mulvenna, William Carswell, Paul McCullagh, Juan Carlos Augusto, Huiru Zheng, Paul Jeffers, Haiying Wang, and Suzanne Martin. Visualization of data for ambient assisted living services. *Communications Magazine, IEEE*, 49(1):110–117, 2011.

[12] Nubia M Gil, Nicolas A Hine, John L Arnott, Julienne Hanson, Richard G Curry, Telmo Amaral, and Dorota Osipovic. Data visualisation and data mining technology for supporting care for older people. In *Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, pages 139–146. ACM, 2007.

[13] Saskia Robben, Mario Boot, Marije Kanis, and Ben Kröse. Identifying and visualizing relevant deviations in longitudinal sensor patterns for care professionals. In *Proceedings of the 7th International Conference on Pervasive Computing Technologies for Healthcare*, pages 416–419. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2013.

[14] Katerina Vrotsou, Kajsa Ellegard, and Matthew Cooper. Everyday life discoveries: Mining and visualizing activity patterns in social science diary data. In *Information Visualization, 2007. IV'07. 11th International Conference*, pages 130–138. IEEE, 2007.

[15] Yunyoung Nam, Seungmin Rho, and S Lee. Extracting and visualising human activity patterns of daily living in a smart home environment. *IET communications*, 5(17):2434–2442, 2011.

[16] Shuang Wang and M. Skubic. Density map visualization from motion sensors for monitoring activity level. In *Intelligent Environments, 2008 IET 4th International Conference on*, pages 1–8, July 2008.

[17] Shuang Wang, Marjorie Skubic, and Yingnan Zhu. Activity density map dis-similarity comparison for eldercare monitoring. In *Engineering in Medicine and Biology Society, 2009. EMBC 2009. Annual International Conference of the IEEE*, pages 7232–7235. IEEE, 2009.

[18] Nicholas M Boers, David Chodos, Jianzhao Huang, Pawel Gburzynski, Ioanis Nikolaidis, and Eleni Stroulia. The smart condo: Visualizing independent living environments in a virtual world. In *Pervasive Computing Technologies for Healthcare, 2009. PervasiveHealth 2009. 3rd International Conference on*, pages 1–8. IEEE, 2009.

[19] Diane J Cook, Aaron S Crandall, Brian L Thomas, and Narayanan C Krishnan. Casas: A smart home in a box. *Computer*, 46(7), 2013.

[20] Brian L Thomas and Aaron S Crandall. A demonstration of pyviz, a flexible smart home visualization tool. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*, pages 304–306. IEEE, 2011.

[21] Chao Chen and Prafulla Dawadi. Casasviz: Web-based visualization of behavior patterns in smart environments. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*, pages 301–303. IEEE, 2011.

[22] M. Chen, D. Ebert, H. Hagen, R.S. Laramee, R. van Liere, K.-L. Ma, W. Ribarsky, G. Scheuermann, and D. Silver. Data, information, and knowledge in visualization. *Computer Graphics and Applications, IEEE*, 29(1):12–19, Jan 2009.

[23] Stephen Few. Data visualization for human perception. *The Encyclopedia of Human-Computer Interaction, 2nd Ed.*, 2013.

[24] Colin Ware. *Information visualization: perception for design*. Elsevier, 2012.

[25] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996.

[26] Imad Afyouni, Ray Cyril, and Claramunt Christophe. Spatial models for context-aware indoor navigation systems: A survey. *Journal of Spatial Information Science*, 1(4):85–123, 2012.

[27] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2301–2309, 2011.