Situvis: A Visual Tool for Modeling a User's Behaviour Patterns in a Pervasive Environment

Adrian K. Clear, Ross Shannon, Thomas Holland,
Aaron Quigley, Simon Dobson, and Paddy Nixon

Systems Research Group
UCD Dublin, Ireland
adrian.clear@ucd.ie

Abstract. One of the key challenges faced when developing context-aware pervasive systems is to capture the set of inputs that we want a system to adapt to. Arbitrarily specifying ranges of sensor values to respond to will lead to incompleteness of the specification, and may also result in conflicts, when multiple incompatible adaptations may be triggered by a single user action. We posit that the ideal approach combines the use of past traces of real, annotated context data with the ability for a system designer or user to go in and interactively modify the specification of the set of inputs a particular adaptation should be responsive to. We introduce Situvis, an interactive visualisation tool we have developed which assists users and developers of context-aware pervasive systems by visually representing the conditions that need to be present for a situation to be triggered in terms of the real-world context that is being recorded, and allows the user to visually inspect these properties, evaluate their correctness, and change them as required. This tool provides the means to understand the scope of any adaptation defined in the system, and intuitively resolve conflicts inherent in the specification.

1 Introduction

Context-aware pervasive systems are designed to support a user's goals by making adaptations to their behaviours in response to the user's activities or circumstances. The accuracy and utility of these adaptations is predicated on the system's ability to capture and recognise these circumstances as they occur. We system designers characterise these adaptation opportunities by collecting context data from multiple heterogeneous sensors, which may be networked physical instruments in the environment (measuring factors like temperature, humidity or noise volume), software sensors retrieving information from the web or various data feeds, or wearable sensors measuring factors like acceleration or object use. These context data are voluminous, highly multivariate, and constantly being updated as new readings are recorded.

Situations are high-level abstractions of context data, which free the user from having to deal with raw context and allow more expressive adaptations [1]. We define situations in terms of context that has been encapsulated to a level of

H. Tokuda et al. (Eds.): Pervasive 2009, LNCS 5538, pp. 327–341, 2009.

understanding appropriate for a developer specifying a situation (e.g., symbolic locations), rather than the raw sensor readings (e.g., 3D coordinates). Situations are straightforward for both system designers and system users to work with, as they symbolically define commonly-experienced occurrences such as a user "taking a coffee break", or being "in a research meeting", without requiring the user to understand any of the dozens of distinct sensor readings that may have gone into making up these situations. Situations are thus a natural view of a context-aware system, while the individual pieces of context are each "a measurable component of a given situation" [2].

Thomson et al. observe that there are two approaches to situation determination: specification-based and learning-based approaches [3]. The specification-based approach suffers from complexity. As the context information available to a context-aware system at any moment is so extensive, dynamic and highly dimensional, it is a significant challenge for a system observer to ascribe significance to changes in the data or identify emergent trends, much less capture the transient situations that are occurring amid the churn of the data.

On the other hand, learning-based approaches require training data and interpretation. Many situations a user finds themselves in are subjective and hence require a degree of personalisation. Here, we propose a hybrid of these user-driven and data-driven approaches that utilises minimal annotated samples to frame a situation specification, combined with a novel visualisation that simplifies the manual process of fine-tuning.

Situvis is our scalable visualisation tool for illustrating and evaluating the makeup of situations in a context-aware system. By incorporating real situation traces and annotations as ground truth, Situvis assists system developers in constructing and evaluating sound and complete situation specifications by essentially bootstrapping the manual process, affording the developer a better understanding of the situation space, and the reliability of modeling with situations based on real, recorded sensor data. It is a framework that allows developers to understand, at a high level, how their system will behave given certain inputs.

The following section provides some details of other approaches to the recognition of context abstractions, a formal description of situation specifications and a review of some challenges faced when working with context and situations. We then describe the details of the Situvis tool, including a demonstration of its utility, followed by a discussion of its properties.

2 Background

2.1 Activity Recognition

Techniques for activity recognition use machine learning techniques—both supervised and unsupervised—to infer high-level activities from low-level sensor data. Logan et al. present a long-term experiment of activity recognition in a home setting using a semi-supervised technique [4]. Like the majority of activity recognition, the focus is on concepts that can be described and recognised

by body movements and object use. 104 hours of video data was manually annotated following the data collection. Many activities, such as dishwashing and meal preparation, were accurately classified to a high degree. However, the study showed that even with this large amount of data and annotation, some activities, such as drying dishes, could not be learned effectively due to lack of training data caused by their infrequent occurrence, even over a 104-hour period.

Krause et al. describe an approach to learning context-dependent personal preferences using machine learning techniques to refine the behaviour of Sensay, a context-aware mobile phone [5]. The behaviour modifications, such as changing the state of the ringer volume from loud to silent, are known in advance. The task is to find the user's "state" (or contextual circumstances) that corresponds to them modifying the behaviour of their phone so that in the future it can be done automatically. Machine learning of personalised states is favoured over manual specification of general states as a result of a study showing that states and desired phone behaviour differed among individuals. Because the behaviour modifications are known, this method requires no supervision. Essentially, the behaviour modifications serve as labels for the recorded sensor values.

Recognising higher-level abstractions. Recent work has aimed to recognise high-level abstractions of context called routines [6]. Routines are structured as compositions of several activities that may be influenced by time, location and the individual performing them. Examples include "commuting" or "working". In contrast to activities, routines cannot be identified through their local physical structure alone: they consist of variable patterns of multiple activities; they range over longer periods of time; and they often vary significantly between instances. Moreover, they are subjective. As a result, the authors chose topic maps as an alternative approach for recognition. Topic maps are a family of probabilistic models often used by the text processing community and enable the recognition of daily routines as a composition of activity patterns.

We too aim to be able to recognise high-level abstractions, and our approach is designed to achieve this with minimal annotation. Situations and routines are similar in that they are subjective, making long periods of annotation unscalable; and they require more factors to recognise them than simply body posture, body movement, or object use. Therefore, accurate situation determination cannot rely completely on data-driven techniques. Situations are generally short term and hence are logically more complex than routines—they can be partially described in terms of individual activities but they are not lengthy nor activity-rich enough to be represented as the most probable activities that are occurring over a long time window. As a result, we are taking a hybrid approach to recognition that includes a short ground truth collection period followed by manual fine-tuning by a domain expert.

2.2 Situation Specifications

Based on the extensive literature on the subject of modeling context for adaptive systems [1,2,7,8,9,10], we can make some observations: the incoming sources of

context into a pervasive application are viewed as a finite number of variables: either nominal or categorical values, e.g., activity levels {idle, active, highly active ... }; or quantitative ordinal values which may be defined over some known interval, e.g., noise level in decibels {0, 140}.

Location information will typically arrive as individual values for an object's x, y and z coordinates in a space, and may be recorded by numerous disparate positioning systems, but is modeled as a higher-level abstraction to make it easier to reason with. Previously conducted research allows component x, y and z coordinates to be composed into a symbolic representation, given some domain information [11], and so we can work with locations as readable as "Simon's office" or "Coffee Area". Our visualisation tool works equally well with simple quantitative data or these higher-order categorised data.

Situations are high-level abstractions that serve as a suitable model with which to develop context-aware systems, because they are intuitive concepts for both designers and users to think in. In order for a computing system to be able to recognise situations, they must first be specified in some way. Theory on the semantics of situation specification can be seen in the work of Henricksen [1] and Loke [12]. Based on this work, we also model situations using declarative languages, which can simply be plugged-in to our tool.

Situation specifications are boolean expressions (or assertions)—they are either true or false, denoting occurrence and non-occurrence, respectively. Assertions may be composed using the logical operators AND ($\wedge$), OR ($\vee$), and NOT ($\neg$), resulting in richer expressions. Domain-specific functions can also be defined to enrich specification semantics (e.g., a distance operator could return a numerical value of the distance between two locations).

We can thus define a situation specification as a concatenation of one or more assertions about contexts, which leads us to the following formal definition:

A situation specification consists of one or more assertions about context that are conjoined using the logical operators AND *($\wedge$),* OR *($\vee$), and* NOT *($\neg$). Assertions may comprise further domain-specific expressions on context, given that the required semantics are available.*

2.3 Interactive Machine Learning

Existing work has applied the coupling of data- and user-driven processes to carry out difficult tasks. In particular, the general Interactive Machine Learning (IML) model consists of iterations of classical machine learning followed by refinement through interactive use. In the Crayons project by Fails and Olsen [13], users can build classifiers for image-based perceptual user interfaces using a novel IML model that involves iterative user interaction in order to minimise the feature set and build a decision tree. Moreover, Dey's *a CAPpella* is a prototyping environment, aimed at end-users, for context-aware applications [14]. It uses a *programming by demonstration* approach, through a combination of machine-learning and user interaction, to allow end-users to build complex context-aware applications without having to write any code.

2.4 Visualisation of Context Data

The field of visual analytics uses interactive visual interfaces to aid end-users in analysing and understanding large and complex multivariate data sets. Interactive visualisation tools help the viewer perform visual data analysis tasks: exploring patterns and highlighting and defining filters over interesting data. For example, Andrienko et al. developed a toolset for analysing and reasoning about movement data (e.g., GPS coordinates). Following some preprocessing steps, the data can be clustered according to different properties, such as start and end points of trips, or similar behaviour over time [15]. Such properties can be portrayed using different types of visualisations to increase user understanding.

There exist myriad visualisation techniques, from time-series to multi-dimensional scatter plot methods, which can be adapted to the exploration of multi-dimensional context data. Our focus here is not only on the exploration of such context data, but also the scope of the higher order situations, their specification, and data cases which fall outside the set boundaries. The Table Lens, a focus+context visualisation, supports the interactive exploration of many data values in a semi-familiar spreadsheet format [16]. In practice, due to the distortion techniques employed, users can see 100 times as many data items within the same screen space as compared with a standard spreadsheet layout. Rather than showing the detailed numeric values in each cell, a single row of pixels, relating to the value in the cell, is shown instead. The Table Lens affords users the ability to easily study quantitative data sets, but categorical values are not well supported.

3 Parallel Coordinates

Parallel Coordinate Visualisations (PCVs) are a standard two-dimensional technique ideally suited to large, multivariate data sets [17]. The technique excels at visually clustering cases that share similar attribute values across a number of independent discrete or continuous dimensions, as they can be visually identified through the distribution of case lines within the visualisation [18]. The user can see the full range of the data's many dimensions, and the relative frequencies at which values on each axis are recorded. These features are visible in Figure 1, which shows context data from our user study, which we will describe in the next section.

PCVs give users a global view of trends in the data while allowing direct interaction to filter the data set as desired. A set of parallel vertical axes are drawn, which correspond to attributes of the readings in the system. Then, a set of n-dimensional tuples are drawn as a set of *polylines* which intersect each axis at a certain point, corresponding to the value recorded for that attribute. Discrete and quantitative axes can be presented in the same view.

As all the polylines are being drawn within the same area, the technique scales well to large data sets with arbitrary numbers of attributes, presenting a

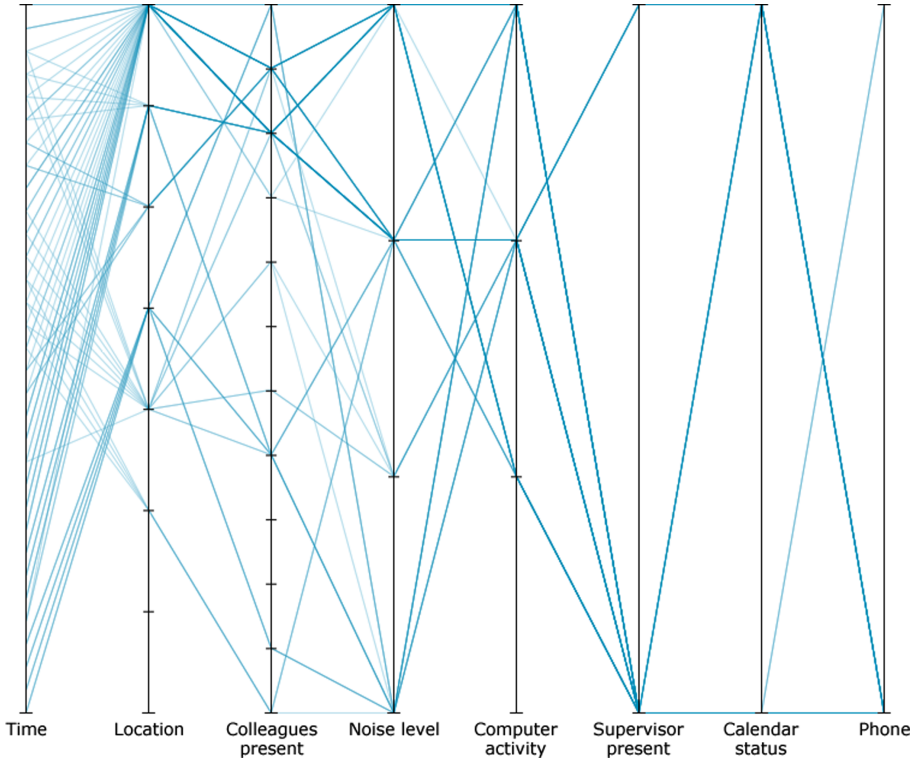

Fig. 1. Part of the main Situvis window showing our Parallel Coordinates Visualisation. This is a view of 96 overlaid context traces with 8 data dimensions gathered over three days of summer. Strong correlations can be seen between the days recorded: the subject spent the majority of all three days at their desk (the first value on the "Location" axis), with some deviations due to coffee breaks or visits to their supervisor's office at irregular times.

compact view of the entire data set. Axes can be easily appended or removed from the visualisation as required by the dimensions of the data.

As Parallel Coordinates have a tendency to become crowded as the size of the data set grows larger, techniques have been designed to cluster or elide sub-sets of the data to allow the dominant patterns to be seen [19]. Direct interaction to filter and highlight sections of the data encourages experimentation to discover additional information, as seen in Figure 2.

Hierarchical clustering [20] uses colour to visually distinguish cases that share a certain range of values into a number of sets, increasing the readability of the diagram. We use a similar technique to group case lines that are assigned to a certain situation, colour-coding these as a group. Different situations can be colour-coded so that the interplay of the context traces that correspond to them can be easily seen. We will illustrate this ability in the next section.

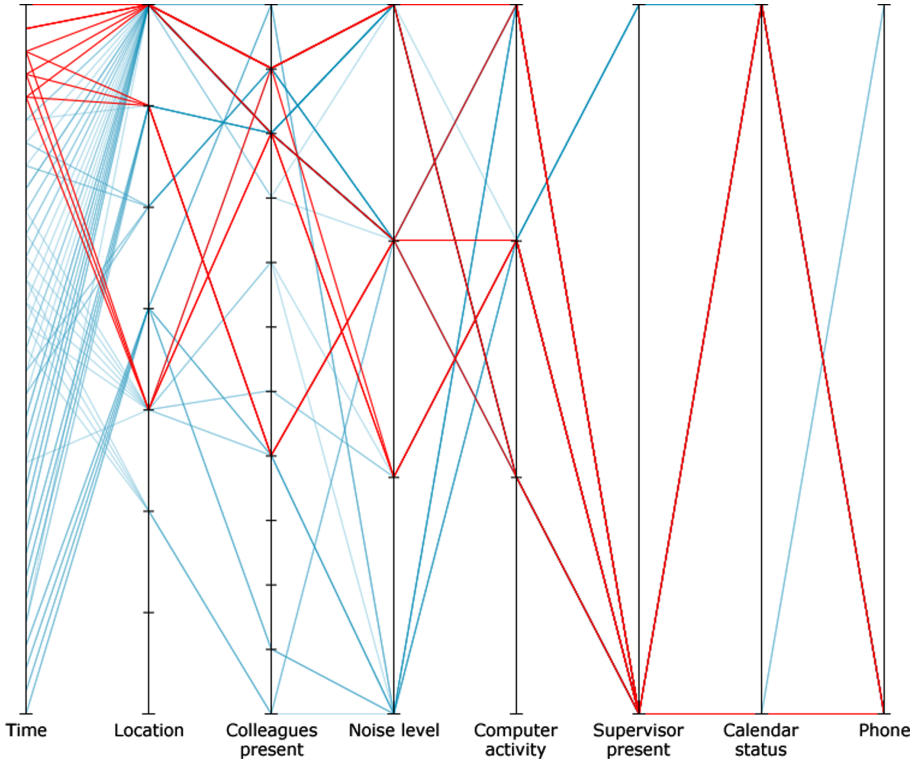

Fig. 2. Here the user has "brushed" over a set of case lines (those that correspond to times before 11am) by right clicking and dragging a line across them between the first and second axes. This highlights these polylines throughout the diagram, allowing the patterns that occurred among these times to be seen. This same operation can be performed on any axis to select any subset of the polylines.

4 Evaluating Situations with Situvis

4.1 Description and Case-Study

The goal of Situvis is to combine data-driven and user-driven techniques for situation determination without relying on machine learning to make sense of the data. The tool displays all of the situation trace data, along with annotations where available, in a single view; and allows the user to clearly and easily highlight the situation traces associated with an annotation label. The user can then adjust the resulting set of ranged intervals over the context to create a more complete and accurate situation specification.

Situvis is built using Processing [21], a Java-based visualisation framework that supports rapid prototyping of visualisation techniques.[1] Each context

[1] Situvis is freely-available software, which you are encouraged to download from our website at `http://situvis.com`.

dimension is represented in Situvis as a vertical axis, and each axis contains a set of points that correspond to permitted values for the dimension. A situation trace is represented as a polyline—a line drawn starting at the leftmost axis and continuing rightwards to the next adjacent and so on, intersecting each axis at the point that represents the value that the context has in that situation trace. For example if, in a given situation, a user's computer activity level is "idle", and their location is "canteen", and these two axes are adjacent, then a line will be drawn between those two points. Each situation trace is plotted on the axes and the result is a view of all of the situations, significant and insignificant, that occurred in the system over a period of time.

To carry out our case-study, we required real context data with which we could characterise situations. We chose to gather context data and situation annotations manually over two three-day periods. While the capabilities exist to collect these context data automatically, for this first trial we chose to collect the data through manual journaling, so that we did not need to factor in issues with the aggregation, uncertainty or provenance of the context data. As mentioned previously, we assume the data is at an appropriate level of abstraction to begin with.

We had a single trial participant record their context every fifteen minutes (between 10am–6pm) for three consecutive weekdays, on two distinct occasions. The first occasion was in summer and the second was in autumn. The journaling gap between these two data sets is designed to capture adjustments in the routines and descriptions of situations that the trial participant found himself in. The captured context consists of time, location, noise-level, number of colleagues present, their supervisor's presence (true or false), their phone use (either taking a call or not), calendar data (being busy or having no appointments), and computer activity. For simplicity, the noise-level was recorded on a 4-point scale of quiet, conversation, chatty, and noisy. Likewise, computer activity level was scaled as idle for an hour or more, idle for less than an hour, active, and highly active. We defined six symbolic locations: meeting room, canteen, sports center, supervisor's office, subject's desk, and a lecture theatre. Figure 1 shows a view of the Situvis tool with all of the traces from the first three consecutive days of collection plotted together in one view.

The participant also annotated what, if any, situation he was in at the time of data capture. These annotations are used in Situvis to identify situations that require specification in the system, and to provide some ground truth to initiate their specification.

4.2 Specifying Situations with Context

Situation specifications are structured according to the definition we discussed in Section 2.2. Situvis enables a developer to select all occurrences of a given annotated situation, and add further cases to this definition using interactive brushing of polylines as in Figure 2, or by dragging a range indicator on the left of the axis to expand or contract the range of values covered by this specification. The user can evaluate existing situation specifications overlaid against actual trace data and see where they need to be modified.

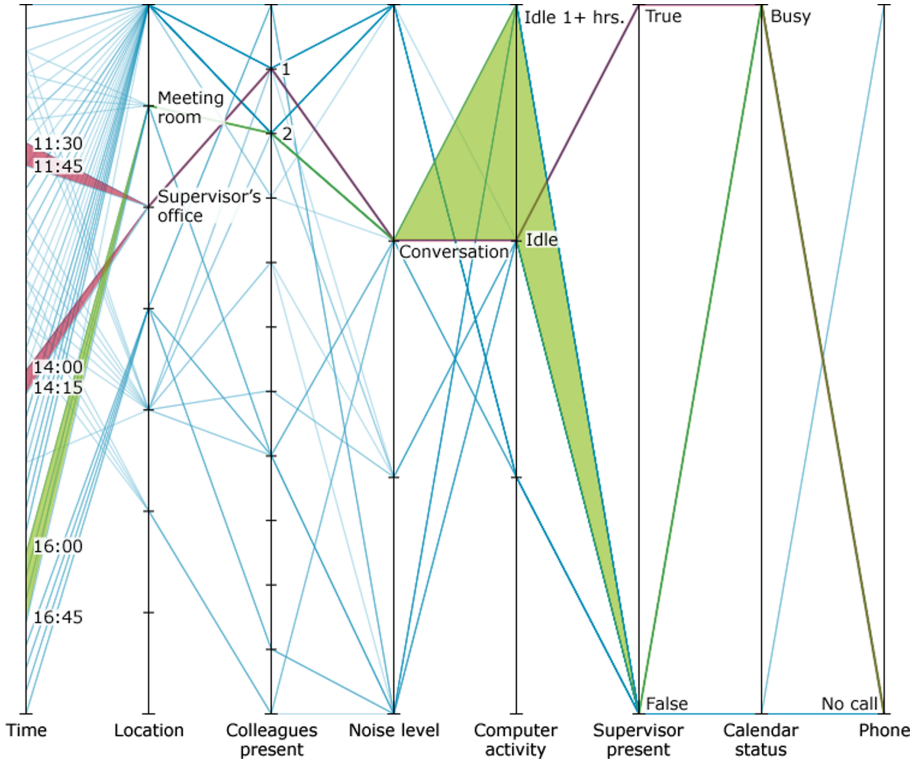

Fig. 3. A view of the Situvis tool with our initial summer data set. The highlighted traces were annotated as a "meeting" situation. These situations occurred at many different times throughout the day in two different locations, with a range of values for the other contexts. Labels have been added to the axis for clarity. They are normally shown when the user hovers over the axis.

An example of this process can be seen in Figure 3 and 4. The trial subject annotated multiple occurrences of a "Meeting" situation [2]. By selecting these traces, it is evident what context dimensions characterise them. We can see that "Time" and "Supervisor presence" are not useful due to the multiple split lines on their axes, and so are ineffective when defining constraints. The specification is clear from the other dimensions, however, and could be expressed as:

$$\{\text{Location} = (\text{Meeting room} \vee \text{Supervisor's office})\} \wedge$$
$$\{1 \leq \text{Colleagues present} \leq 2\} \wedge$$
$$\{\text{Noise-level} = \text{conversation}\} \wedge$$
$$\{\text{Computer activity} \geq \text{idle}\} \wedge$$

[2] "Meeting" is a generalisation of two situations that the participant labelled, namely, "Meeting with colleagues" and "Meeting with supervisor". These are assigned separate colours in the figures.

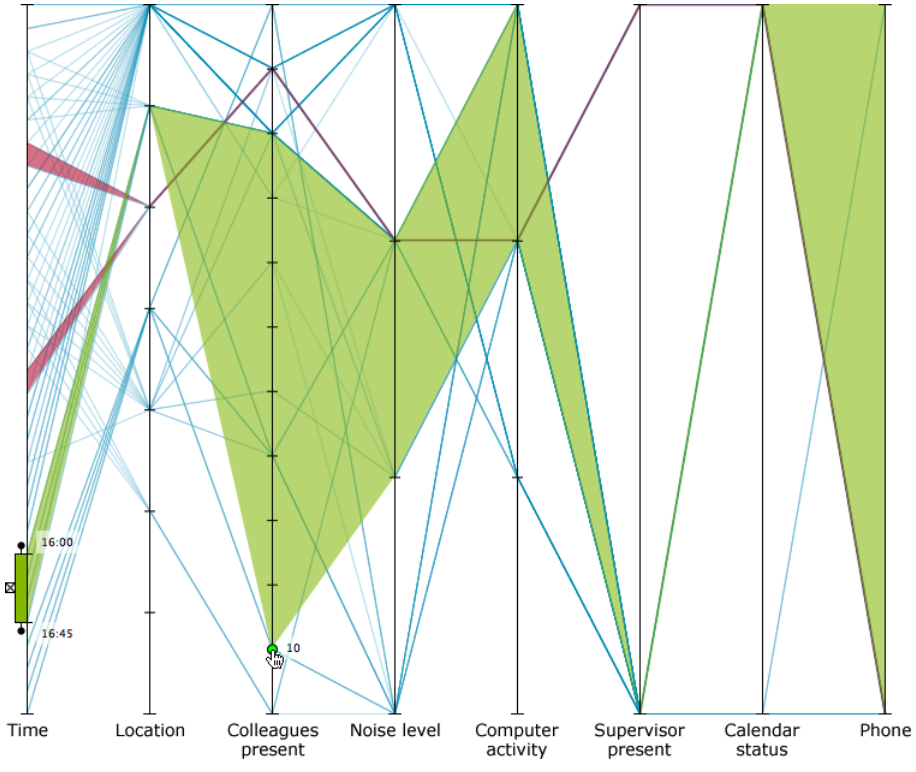

Fig. 4. The user can interactively expand or contract the situation definition along any of the axes. In this case, they have chosen to modify the situation specification to allow for more colleagues to be present, the noise level to be greater and the possibility of talking on the phone.

$\{\text{Calendar status} = \text{busy}\} \wedge$
$\{\text{Phone use} = \text{none}\}$

None of these values alone can characterise "Meeting", as the trace data illustrates. Furthermore, each dimension may not always be available. Situvis allows one to identify combinations of dimensions which, when taken together can provide a good estimation of the situation. For example, "Location" taken with "Colleagues present" is a good indication of "Meeting". This can also give system developers an insight into which sensors in their system are the most useful, and which types of sensors they should invest in to gain the most added benefit in terms of the expressiveness of their system.

4.3 Situation Evolution

When existing specifications are overlaid on the trace polylines, the developer can see where they are too strong or weak. Constraints that are too strong will cause the system to sometimes fail in determining when that situation is occurring.

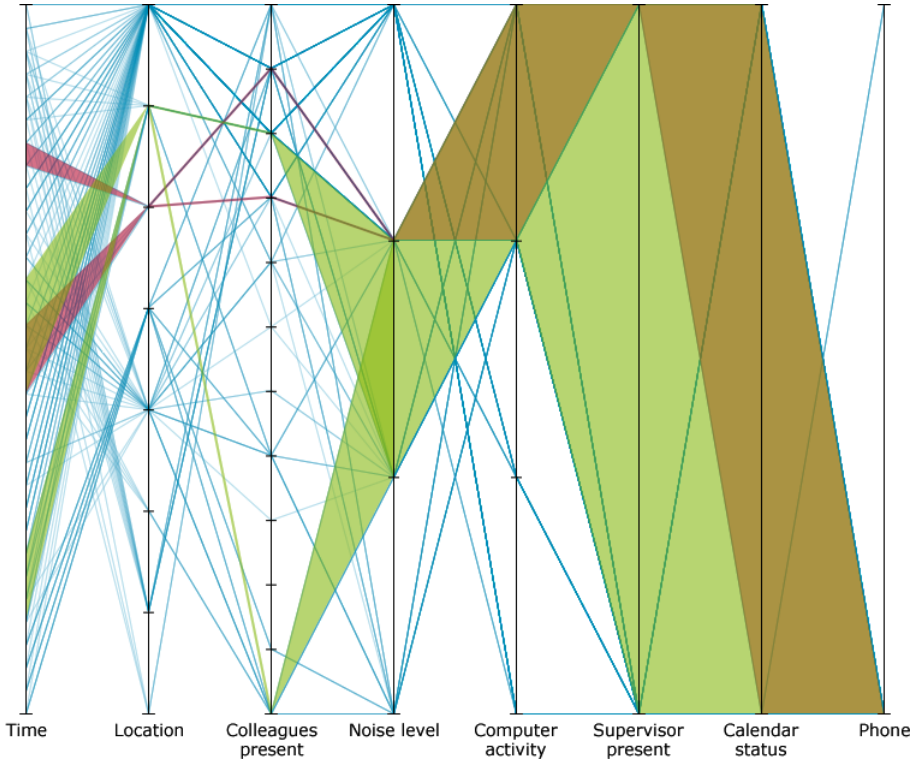

Fig. 5. After additional data collection, the case lines annotated as "Meeting" exhibit a different pattern

Constraints that are too weak may be wrongly interpreted as an occurrence of the specified situation, when in fact a different situation is occurring. By overlaying our specification on top of the polylines, it will be obvious where constraints need to be strengthened, weakened or even excluded altogether. Situvis enables a developer to drag the boundaries of specifications to change the polylines that they cover, essentially changing the constraints of the situation.

When the overlaid situation encompasses traces that are not relevant, the user can strengthen the constraints by narrowing the range of values covered by this situation specification (the shaded area in the diagram). Similarly, the user can weaken constraints to include traces that happen to fall outside the existing specification by widening the specification, as we have done in Figure 4.

As more trace data is added and annotated, the constraints that we have defined for "Meeting" may be shown to be too strong. This is what we found to be the case in Figure 5, which contains the traces for both our initial summer data set, as well as the additional days from the autumn data set, for a total of 48 hours of context traces. What we see is that in most cases, the previously apparent patterns are strengthened, as essentially they have recurred. Comparing Figure 3 and Figure 5, we can see that the annotated data that the

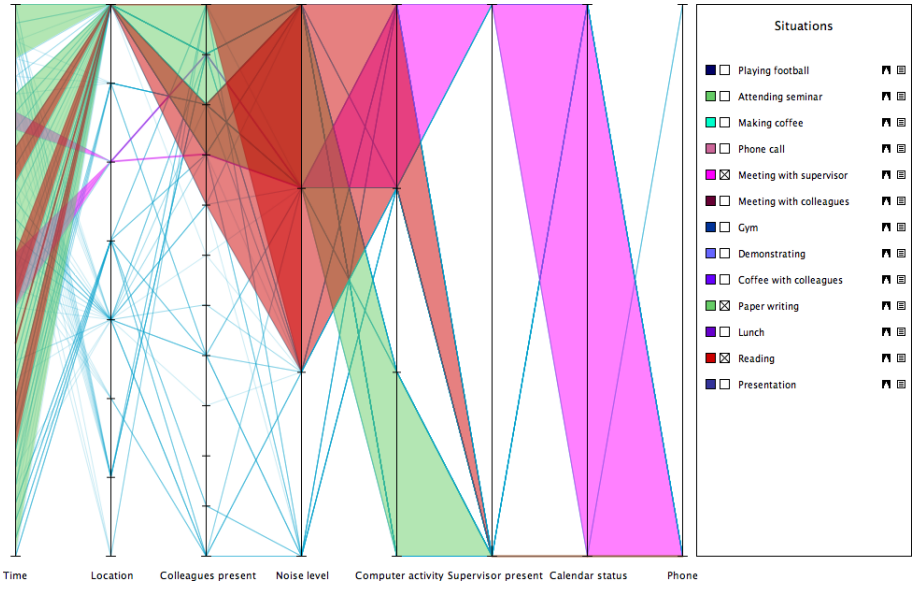

Fig. 6. A view of three distinct situations. Here we are showing the specifications for a meeting with supervisors, paper writing time, and time spent reading. The dissimilarities between these situations are clear from the tool, and the specifications can be further teased apart if required.

user has defined as corresponding to meetings results in a different situation specification.

4.4 Situation Evaluation

Context-aware adaptive systems are very sensitive to incompatible behaviours. These are behaviours that conflict, either due to device restrictions, such as access to a public display, or due to user experiences, such as activating music playback while a meeting is taking place. Situations are closely tied to behaviours—they define envelopes in which behaviour occurs. As a result, their specifications are directly responsible for adherence to compatibility requirements. By harnessing this factor, we can address another key aspect of situation evaluation.

Conceptually relating situations to each other from a behaviour compatibility standpoint is an overwhelming task for a developer. We recognise that there are two situation relationships that may lead to incompatibility:

subsumption if a subsumes b, and b occurs, then a will certainly occur.
overlap if a overlaps b, then a and b may co-occur.

Our tool allows multiple situation specifications to each be coloured distinctly. When two or more situations are shown together, the overlap in their constituent contexts is clear, as well as the extent of their dissimilarities. This view allows

the developer to alter constraints where necessary, while the overlap and subsumption relationships are refreshed and displayed on-the-fly. A screenshot of this scenario is seen in Figure 6, which also shows the specification selection panel on the right-hand side. This area allows the user to toggle specifications on and off, so that they can be compared and manipulated.

5 Conclusions and Future Work

We have presented Situvis, a tool that uses a Parallel Coordinate Visualisation to illustrate situation traces and specifications. We have shown, using a case-study, the utility of Situvis in the situation specification and evaluation processes. Situvis presents a developer with a reference point for situation specification and evaluation through the display of actual trace data and situation annotations. The relevance of the underlying context to a specification is made clear, and contrasting situation traces can be used as a guide for specification.

Context-aware systems are dynamic—sensors, users and habits are constantly changing. Hence, we cannot expect situation specifications to remain static. It must be possible to re-evaluate them accordingly. Situvis allows developers to visually overlay specifications on traces, and tailor their constraints as a result. Unlike traditional methods, Situvis clearly depicts cases where constraints are too strong or too weak. Machine-learning techniques would require extra time-consuming training periods for the re-evaluation process, whereas Situvis provides the option of collecting a minimal amount of annotated data to initiate the manual process.

By visually analysing the overlap of situation specifications within their system, the developer can identify where multiple situations require similar context values to be activated. Such overlaps may imply problems in the situation specifications, as conflicting behaviours may be triggered by conceptually similar situations. Thus, the developer can compare situations against others, and change the situation's specifications to become stronger or weaker as necessary.

A feature missing from the current version of the Situvis tool is explicit support for probabilities in situation specifications. In many context-aware applications, robust probabilistic inference is a requirement to handle the naturally fuzzy data in the system. We are considering the addition of an overlay which will allow users to set up a probability distribution, though this requires a more in-depth study of the treatment of uncertainty in situations.

Some context dimensions are not easily represented on a line. In particular, Location within buildings, with its domain relations like subsumption, is difficult to represent in two dimensions. We are researching techniques to flatten hierarchies for a more intuitive representation of this context.

Situvis could also be used by users of the context-aware system as a gateway to user programming: helping them to unroll the cause of a situation activation, so that they can gain insight into why the system began to behave as it did.

Acknowledgments. This work is partially funded under The Embark Initiative of the Irish Research Council for Science, Engineering and Technology, and by Science Foundation Ireland under grant number 03/CE2/I303-1, "LERO: the Irish Software Engineering Research Centre."

References

1. Henricksen, K.: A Framework for Context-Aware Pervasive Computing Applications. PhD thesis, The School of Information Technology and Electrical Engineering, University of Queensland (September 2003)
2. Knox, S., Clear, A.K., Shannon, R., Coyle, L., Dobson, S., Quigley, A., Nixon, P.: Towards Scatterbox: a context-aware message forwarding platform. In: Fourth International Workshop on Modeling and Reasoning in Context in conjunction with Context 2007, Roskilde, Denmark, pp. 13–24 (August 2007)
3. Thomson, G., Stevenson, G., Terzis, S., Nixon, P.: A self-managing infrastructure for ad-hoc situation determination. In: Smart Homes and Beyond - ICOST2006 4th International Conference On Smart Homes and Health Telematics. Assistive Technology Research Series, pp. 157–164. IOS Press, Amsterdam (2006)
4. Logan, B., Healey, J., Philipose, M., Tapia, E.M., Intille, S.: A Long-Term Evaluation of Sensing Modalities for Activity Recognition. In: Krumm, J., Abowd, G.D., Seneviratne, A., Strang, T. (eds.) UbiComp 2007. LNCS, vol. 4717, pp. 483–500. Springer, Heidelberg (2007)
5. Krause, A., Smailagic, A., Siewiorek, D.P.: Context-aware mobile computing: Learning context-dependent personal preferences from a wearable sensor array. IEEE Transactions on Mobile Computing 5(2), 113–127 (2006)
6. Huýnh, T., Fritz, M., Schiele, B.: Discovery of activity patterns using topic models. In: UbiComp 2008: Ubiquitous Computing, 10th International Conference, Seoul, South Korea, pp. 1–10 (September 2008)
7. Clear, A.K., Knox, S., Ye, J., Coyle, L., Dobson, S., Nixon, P.: Integrating multiple contexts and ontologies in a pervasive computing framework. In: C&O 2006: ECAI 2006 Workshop on Contexts and Ontologies: Theory, Practice and Applications, Riva Del Garda, Italy, pp. 20–25 (August 2006)
8. Coutaz, J., Rey, G.: Foundations for a theory of contextors. In: CADUI: 4th International Conference on Computer-Aided Design of User Interfaces, pp. 13–34. Kluwer, Valenciennes (2002)
9. Dey, A.K.: Understanding and using context. Personal Ubiquitous Computing 5(1), 4–7 (2001)
10. Coutaz, J., Crowley, J., Dobson, S., Garlan, D.: Context is key. Communications of the ACM 48(3), 49–53 (2005)
11. Ye, J., Coyle, L., Dobson, S., Nixon, P.: A unified semantics space model. In: Hightower, J., Schiele, B., Strang, T. (eds.) LoCA 2007. LNCS, vol. 4718, pp. 103–120. Springer, Heidelberg (2007)
12. Loke, S.W.: Representing and reasoning with situations for context-aware pervasive computing: a logic programming perspective. The Knowledge Engineering Review 19(3), 213–233 (2004)
13. Fails, J., Olsen, D.: A design tool for camera-based interaction. In: CHI 2003: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 449–456. ACM Press, New York (2003)

14. Dey, A.K., Hamid, R., Beckmann, C., Li, I., Hsu, D.: A cappella: programming by demonstration of context-aware applications. In: CHI 2004: Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 33–40. ACM Press, New York (2004)
15. Andrienko, G., Andrienko, N., Wrobel, S.: Visual analytics tools for analysis of movement data. SIGKDD Explorations Newsletter: Special issue on visual analytics 9(2), 38–46 (2007)
16. Tenev, T., Rao, R.: Managing multiple focal levels in table lens. In: Infovis 1997: IEEE Symposium on Information Visualization, p. 59. IEEE Computer Society Press, Los Alamitos (1997)
17. Inselberg, A., Dimsdale, B.: Parallel coordinates: a tool for visualizing multidimensional geometry. In: VIS 1990: Proceedings of the 1st conference on Visualization 1990, pp. 361–378. IEEE Computer Society Press, Los Alamitos (1990)
18. Card, S., Mackinlay, J., Schneiderman, B.: Readings in Information Visualization: Using Vision to Think. Morgan Kaufmann, San Francisco (1999)
19. Artero, A., de Oliveira, M., Levkowitz, H.: Uncovering clusters in crowded parallel coordinates visualizations. In: IEEE Symposium on Information Visualization, pp. 81–88 (2004)
20. Fua, Y.H., Ward, M.O., Rundensteiner, E.A.: Hierarchical parallel coordinates for exploration of large datasets. In: VIS 1999: Proceedings of the conference on Visualization 1999, pp. 43–50. IEEE Computer Society Press, Los Alamitos (1999)
21. Reas, C., Fry, B.: Processing: a learning environment for creating interactive web graphics. In: SIGGRAPH 2003: ACM SIGGRAPH 2003 Sketches & Applications, p. 1. ACM Press, New York (2003)