QBP Notation for Explicit Representation of Properties, Their Refinement and Their Potential Conflicts: Application to Interactive Systems

Camille Fayollas, Célia Martinie, Philippe Palanque, Yamine Aït-Ameur

QBP Notation for Explicit Representation of Properties, their Refinement and their Potential Conflicts: Application to Interactive Systems

Camille Fayollas, Célia Martinie, Philippe Palanque[1], and Yamine Ait-Ameur[3], FORMEDICIS[2]

[1] ICS-IRIT, Université Toulouse III, France
[2] Collective name
[3] ACADIE-IRIT, ENSEEIHT, France

{fayollas, martinie, palanque, yamine}@irit.fr

Abstract. This paper presents a notation called QBP (Question, Behavior, Property) to represent software and system properties and their relationship. The properties are structured in a tree-shape format from very abstract and generic ones (such as safety or security) to more concrete (leave of the tree). This tree-shape representation is used in the paper to represent properties classification in several areas such as Dependable and Secure computing and Human-Computer Interaction. The notation makes it possible to connect the properties among each other and to connect them to concrete properties expressed in temporal logic. Those concrete properties are, in turn, connected to behavioral descriptions of interactive systems satisfying (or not) the properties. An example is given on a set of different traffic lights from different countries.

Keywords: Properties, interactive systems, safety, security, usability, user experience.

1 Introduction

With the early work on understanding interactive systems [1] came the identification of properties that "good" interactive systems should exhibit (e.g. honesty) and "bad" properties that they should avoid (e.g. deadlocks). Later, guidelines for the design of interactive systems [22] were provided, identifying in a similar way "good" properties (e.g. guidance), in order to favor usability of these systems. In the area of software engineering, early work [7] identified two main good properties of computing systems namely safety (i.e. nothing bad will ever happen) and liveness (i.e. something good will eventually happen). In [10] a hierarchy of software properties is proposed identifying for the first time explicit relationships between properties gathered in a hierarchy (e.g. "reactivity" divided in "recurrence" and "persistence"). While in the area of Human-Computer Interaction the properties were initially expressed in an informal way, [17], [16] proposed the use of temporal logics to describe these properties.

Beyond these “generic” properties, it might be of interest to represent specific properties related to the very nature of each system. These properties might also be of a high level of abstraction (e.g. trust for a banking system) or of very low level (e.g. only possible to enter a personal identification number 3 times on a cash machine). The detailed property would contribute to the high-level one.

2 The QBP Notation

TEAM notation [6,11,13] is an extension of MacLean and al.’s QOC (Question Option Criteria) [9] that allows the description of available options for a design question and the selection of an option according to a list of criteria. The TEAM notation extends QOC to record the information produced during design meetings. For the purpose of work presented here, we propose a refinement of TEAM to explicitly represent properties and their relations including:

- Questions that have been raised (Square colored in pink in Fig. 1),
- Behavioral representations of a system providing an answer to the related question(s) (Disc coloured in orange in Fig. 1),
- Concrete properties (which could be represented in modal logics) describing a desired property that could be met (or not) by the related behavioral description (Triangle colored in green in Fig. 1),
- Refined properties and Properties that represent a hierarchy of “generic” properties that are desired. (Rectangle-triangle colored in blue in right-hand side of Fig. 1).

QBP models make explicit both the hierarchies of properties (that would be represented on the right-hand side of the models) and the concrete design of a system (represented on the left-hand side of the models). .

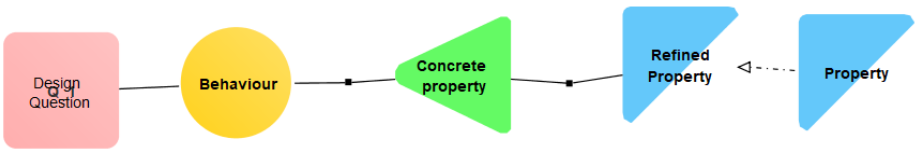


Fig. 1. Main elements of the notation TEAM forming a QBP model

The software tool DREAM [6,11,13] provides support for the editing, recording and analysis of QBP diagrams. In previous work, we have proposed an approach for the selection and management of conflicting guidelines based on the TEAM notation [6, 11, 13]. More specifically, the notation was used for exhibiting choices and trade-offs when combining different guidelines sets. Similar modeling and analysis of models can be performed with QBP.

3 Representing Hierarchies of Properties

This section presents the modeling of several classification of properties using QBP. Some of them are dedicated to interactive systems (see sections 3.1 and 3.3) while other ones are more generic to computing systems (see section 3.2).

The aim is double: first to highlight the fact that the literature has been already proposing hierarchies of properties, second to provide a list of properties dedicated to interactive systems.

3.1 Usability and User Experience

These two major properties in Human-Computer Interaction don't have currently the same level of maturity. Usability has been studied since the early 80's and has been standardized by ISO in the ISO 9241 part 11 since 1996 [5]. Its structure is presented on the a) section of **Fig. 2**. The standard specializes Usability into three sub-properties (efficiency, effectiveness and satisfaction) while some researchers would also add at least Learnability and Accessibility [14] as important aspects of Usability.

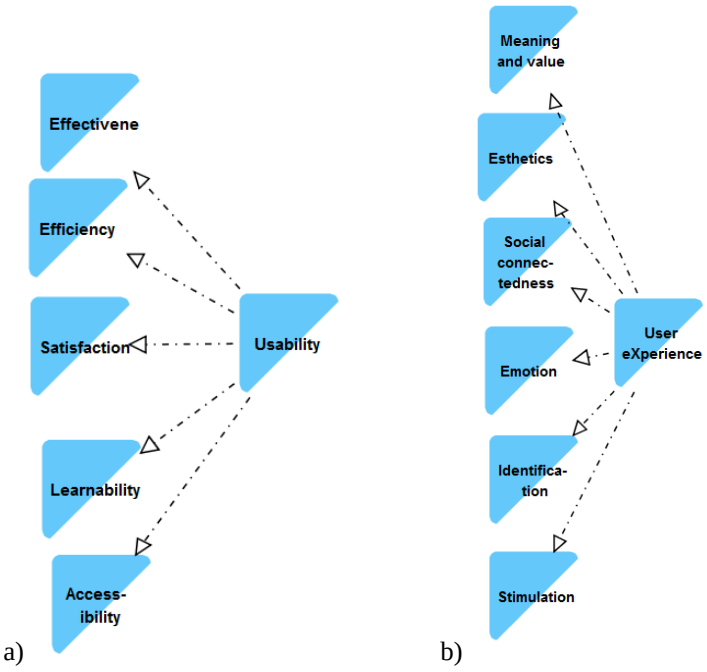


Fig. 2. Representation of the hierarchical relationships between factors and sub-factors of a) Usability [5] and b) User eXperience [15]

User Experience is a more recent concept that is under standardization but still not mature. Sub-properties of User Experience (usually called dimensions) are diverse in terms of level of abstraction and vary widely amongst authors (see [4] for a descrip-

tion of user experience in terms of hedonic and ergonomic qualities – another word for properties). [15] proposes the only set of dimensions that has been carefully check for orthogonality and proposes six dimensions at the same level of abstraction (see right-hand side b) section of **Fig. 2**)

3.2 Dependable and Secure Computing and Concurrent Programs Properties

The first issue of the IEEE transactions on Dependable and secure computing included a paper [8] dedicated to a taxonomy of properties of those systems. The taxonomy is presented in part a) of **Fig. 3**. Beyond a very clear definition of each property this classification shows that some sub-properties such as availability are related to higher-level properties namely safety and security. Indeed, a loss of availability might impact dependability of the systems (if the service not available is requested) while security attacks might target at a reduction of availability of service (as in the classical DDoS – Distributed Denial of Service).

The right-hand side of **Fig. 3** presents a very old and classical decomposition of properties of concurrent systems: safety and liveness that have been introduced in the introduction. Beyond this separation, Sistla proposed in [20] a refinement of these properties in more precise ones contributing to the presence or the absence of the more high-level ones.

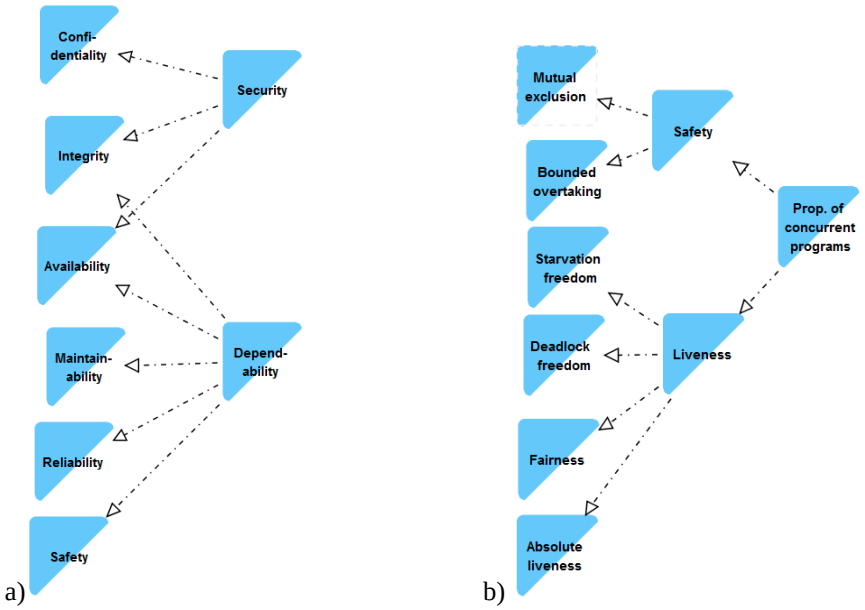


Fig. 3. Representation of hierarchical relationships between factors and sub-factors of Security and Dependability [8] (a) as well as of concurrent programs [16, 17]

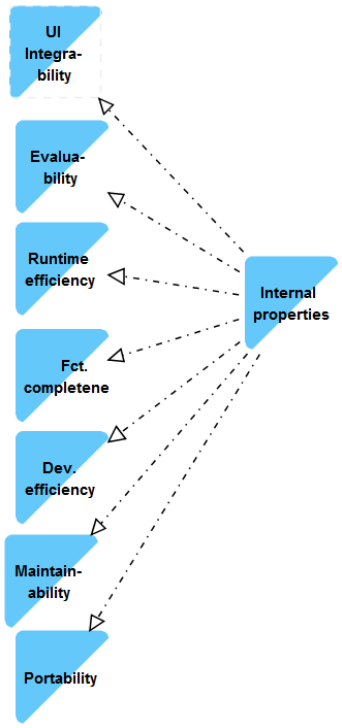


Fig. 4. Representation of hierarchical relationships between factors and sub-factors of Internal properties of user interfaces [2]

3.3 Internal and External Properties of Interactive Systems

In his seminal work in the domain of formal methods for interactive systems [1], Dix proposed a detailed classification of properties in two main groups: external and internal properties. This refers to the fact that part of the interactive system is perceivable by the user and that what is presented to the user might be of "good" quality (which means the presence of the external properties as detailed in **Fig. 5**). The internal properties (see **Fig. 4**) refer to the quality of the interactive system focusing on its internal behavior. These properties are thus closer to the ones presented above in the area of computing systems.

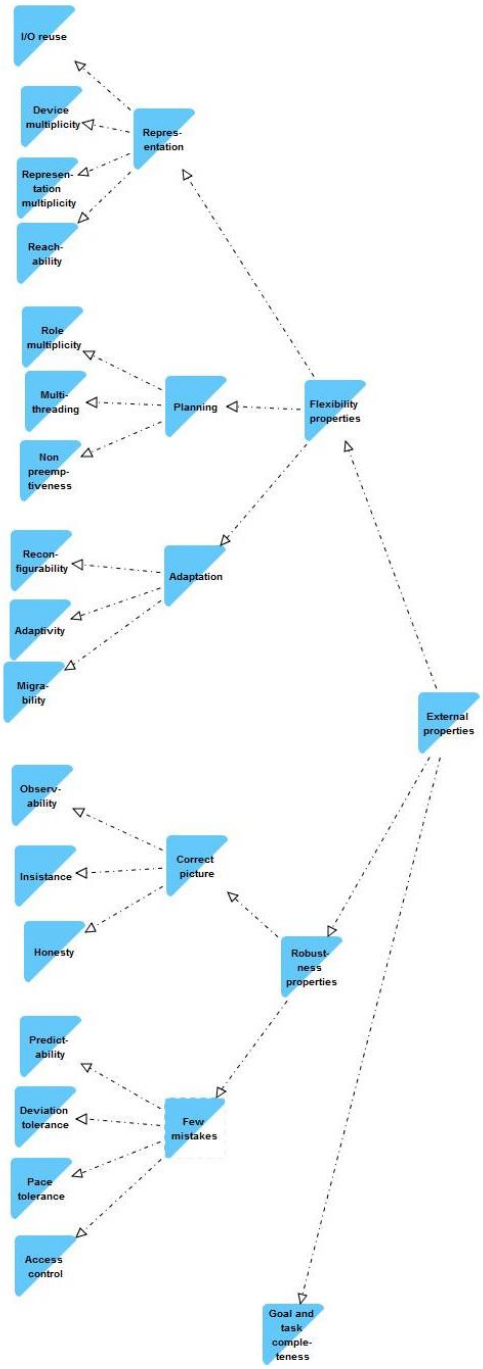


Fig. 5. Representation of hierarchical relationships between factors and sub-factors of External properties of user interfaces [3]

4 The Traffic Lights Case Study

This section presents the application of QBP notation on a simple interactive system. The system has been chosen as it is both simple and widely known so being easily understandable by the reader. It can also serve as a benchmark for other research work on properties descriptions.

4.1 Informal Description of the Case Study

Our case study is an application simulating a traffic light. This application, displayed in Fig. **6**, is made up of three light bulbs (the top one is red (see Fig. 6.b), the middle one is orange (see Fig. 6.c) and the bottom one is green (see Fig. 6.d)). The traffic light exhibits three different modes of operation: i) when it is stopped, ii) when it is working and iii) when it is faulty. In the stopped mode, all the light bulb are switched off (see Fig. 6.a). In the faulty mode, the orange light bulb is blinking (it is switched off during 400 ms and switched on during 600 ms). Finally, the working mode is different following the countries in which it is deployed. We will further details this working mode in the following section for four difference traffic lights: French, British and the Austrian traffic light (for which two different alternatives will be provided).

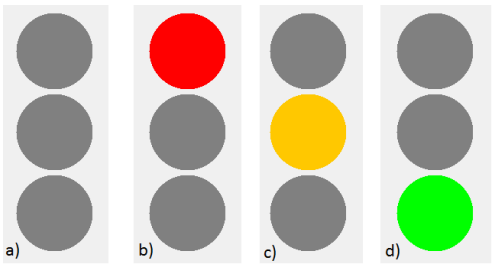


Fig. 6. Screenshots of the traffic light application: a) when it is stopped, b) when the red light bulb is switched on, c) when the orange light bulb is switched on and d) when the green light bulb is switched on.

4.2 Behavioral modelling of the Case Study

This section presents successively the four behavioral models for each of the traffic lights in the case study.

French Traffic light.

Informal Presentation.

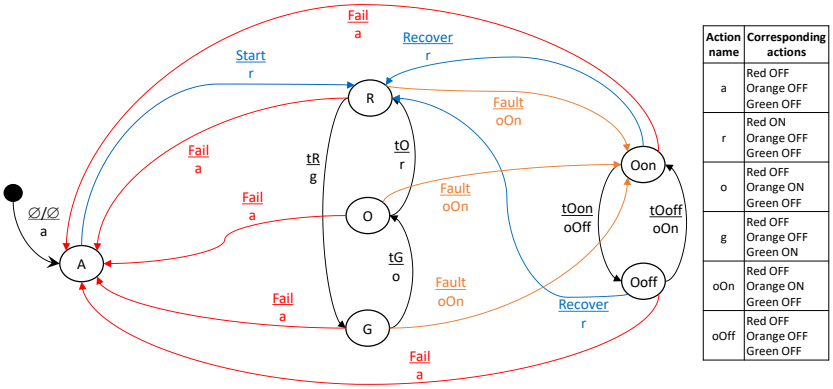
The French traffic light is the simpler one and the other ones are more complex and precise behavior of the French one. When entering the working mode, the traffic light starts with only the red light on, after 1000 ms the red lightbulb is switched off and

the green lightbulb is switched on. This bulb remains on for 2000ms before being switched off while the orange light is switched on for 500ms. When this delay is elapsed, the traffic light comes back to the initial state with only the red light on.

At any time, a fault event may occur that will set the traffic light to the faulty mode. When entering this mode whatever light which is on is switched off and the orange light is switched on for 600 ms (as explained in the informal presentation of the case study above). At any time, a recover event may be triggered setting the traffic light to the initial state of the working mode (i.e. only the red light switched on). A fail event may also occur. When this occurs, whatever state the traffic light is in, it is set to the Fail mode (represented by the state A in **Fig. 7**).

Behavioral model.

Fig. 7 represents with an Augmented Transition Network [24] the behavior described informally above. In the initial state, the traffic light is in the Fail mode (state A in the diagram). When an event Start is received, the traffic light changes state to the R state in the diagram. During this state change, the red lightbulb is switched on ("r" action on the arc label from state "A" to state "R"). From that initial state of the working mode, the timer "tR" will be switched on starting the autonomous behavior of the traffic light in this mode, alternating from Red to Green, from Green to Orange and then back to Red.



Action name	Corresponding actions
a	Red OFF Orange OFF Green OFF
r	Red ON Orange OFF Green OFF
o	Red OFF Orange ON Green OFF
g	Red OFF Orange OFF Green ON
oOn	Red OFF Orange ON Green OFF
oOff	Red OFF Orange OFF Green OFF

Fig. 7. Automaton of the French traffic light

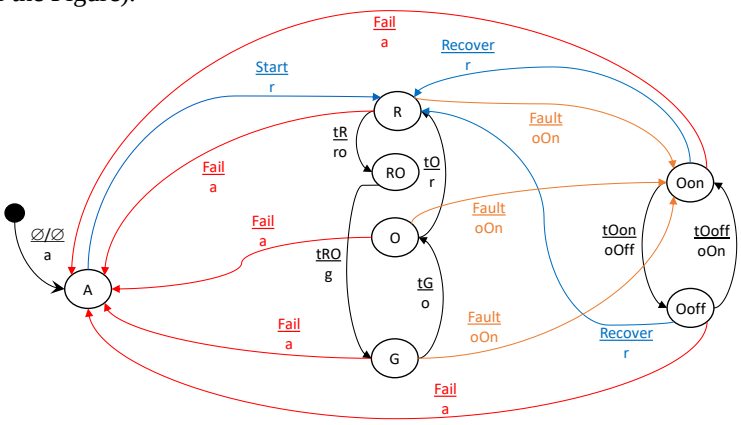
British Traffic light.

Informal Presentation.

Informally, the behavior of the British traffic light is very similar to the French one. The only difference is the fact that, in the working mode, the traffic light does not go directly from Red to Green. An intermediate state has both orange and red lights on before the green lightbulb is switched on. The rest of the behavior (fail and fault modes) remains the same. This behavior makes possible to users to know that the traffic light is going to be green (when both orange and red lights are on).

Behavioral model.

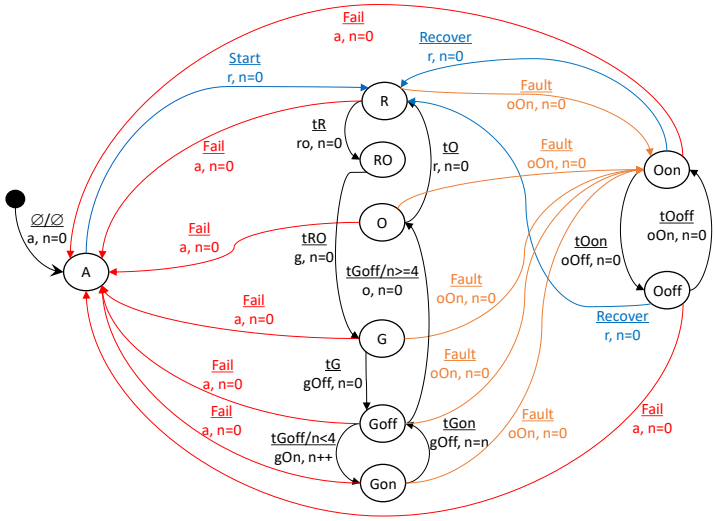
Fig. 8 presents the behavior of the British traffic light. As explained above the only difference is the addition of a stated “RO” between “R” and “G” states (at the center of the Figure).



Action name	Corresponding actions
a	Red OFF Orange OFF Green OFF
r	Red ON Orange OFF Green OFF
ro	Red ON Orange ON Green OFF
o	Red OFF Orange ON Green OFF
g	Red OFF Orange OFF Green ON
oOn	Red OFF Orange ON Green OFF
oOff	Red OFF Orange OFF Green OFF

Fig. 8. Automaton of the British traffic light

Austrian Traffic light.



Action name	Corresponding actions
a	Red OFF Orange OFF Green OFF
r	Red ON Orange OFF Green OFF
ro	Red ON Orange ON Green OFF
o	Red OFF Orange ON Green OFF
g	Red OFF Orange OFF Green ON
gOff	Red OFF Orange OFF Green OFF
gOn	Red OFF Orange OFF Green ON
oOn	Red OFF Orange ON Green OFF
oOff	Red OFF Orange OFF Green OFF

Fig. 9. Automaton of the Austrian traffic light

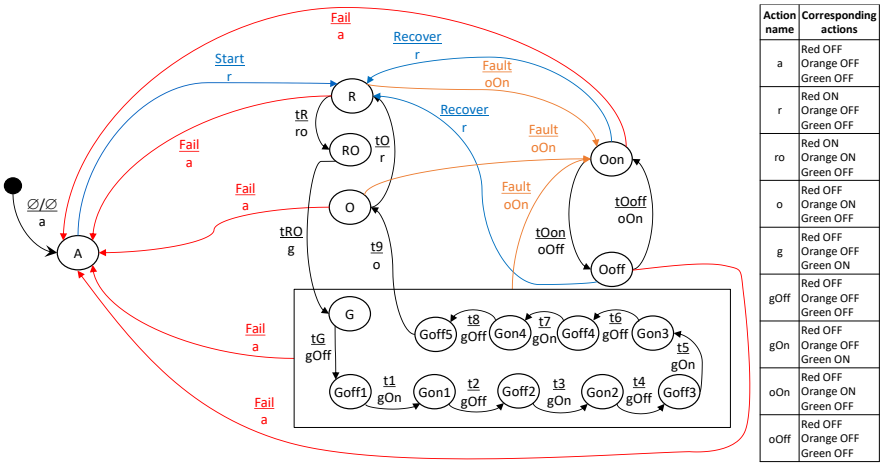
Informal Presentation.

Informally, the Austrian traffic is an extension of the British traffic light. The only difference is when the green light is on. In that state, the Austrian traffic light will present a blinking green status. The green light will blink 4 times before the green light goes definitively off and the orange light is switched on. This allows users to

know that the green light will finish soon and that it is thus better to start to break (or to accelerate in order to avoid being stuck at the red light).

Behavioral model simple.
The model in **Fig. 9** presents one possible description of the behavior presented above. The "G" state in previous models is now a set of three states, the original "G" state plus a set of two states "Goff" and "Gon" modelling the blinking in green light. A timer will alternatively set the automata from state "Goff" to "Gon" until this has been performed the adequate number of time. The number of blinking is stored in the variable (called register in ATNs) n that increases each time the green light is switched on (label n++). When this has been performed 4 times (precondition n>=4 on the label from state "Goff" to "O", the orange light is switched on and the traffic light goes to the state "O".
What is interesting with this model is that it is very easy to increase or decrease the number of times the traffic light will blink green. Indeed, only the values of the two preconditions for the event TGoff have to be changed. Replacing the value 4 by a value 6 would make the traffic light blink six times in the Green blinking mode.

Behavioral model revised.
A revised version of the model above model is presented in **Fig. 10**. It exhibits the same behavior but does not include a precondition to count the number of blinking. Instead these blinking states are unfolded in a number of sequencial Goff and Gon states.



Action name	Corresponding actions
a	Red OFF Orange OFF Green OFF
r	Red ON Orange OFF Green OFF
ro	Red ON Orange ON Green OFF
o	Red OFF Orange ON Green OFF
g	Red OFF Orange OFF Green ON
gOff	Red OFF Orange OFF Green OFF
gOn	Red OFF Orange OFF Green ON
oOn	Red OFF Orange ON Green OFF
oOff	Red OFF Orange OFF Green OFF

Fig. 10. Automaton of the Austrian traffic light revised

He main advantage of this model is that it is very easy change the blinking speed (for instance if we want to represent a faster blinking when the traffic light get closer to state change with orange lightbulb on. However, adding more blinking will deeply change the automata (adding 2 states and 2 timers for each additional blinking.

4.3 Description of properties on the French traffic light

Fig. 11 connects the relevant properties from the literature that have been presented in section 3 with the French traffic light from the case study. A set of 8 concrete properties have been represented that are, in turn, connected to higher-level properties.

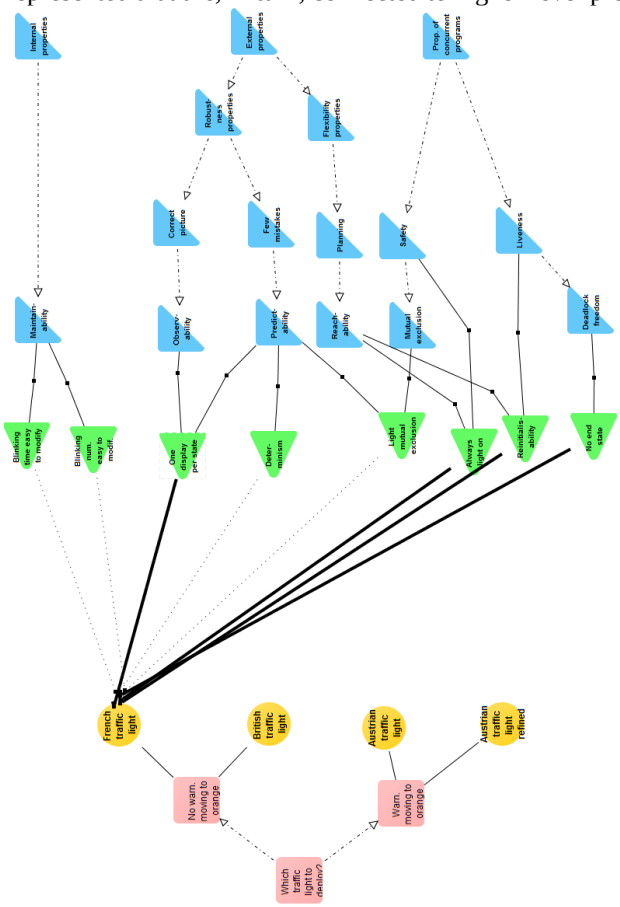


Fig. 11. DREAM diagram for the design options of the traffic light (focus on the relationships between first option and criteria)

The concrete properties are (from top to bottom):

- Blinking time easy to modify
- Blinking number easy to modify
- One display per state
- Determinism
- Light mutual exclusion
- Always at least one light on
- Reinitializability
- No end state

As the French traffic light has no green blinking state, it is not easy to modify the number of blinking nor the speed of blinking. This is why the relationship between the behavior of the French traffic light and these properties is a dashed line (meaning that the property is not true with this model). These dashed and bold lines were previously used in QOC [9] to represent the fact that a given option (orange circle) favors a given criteria.

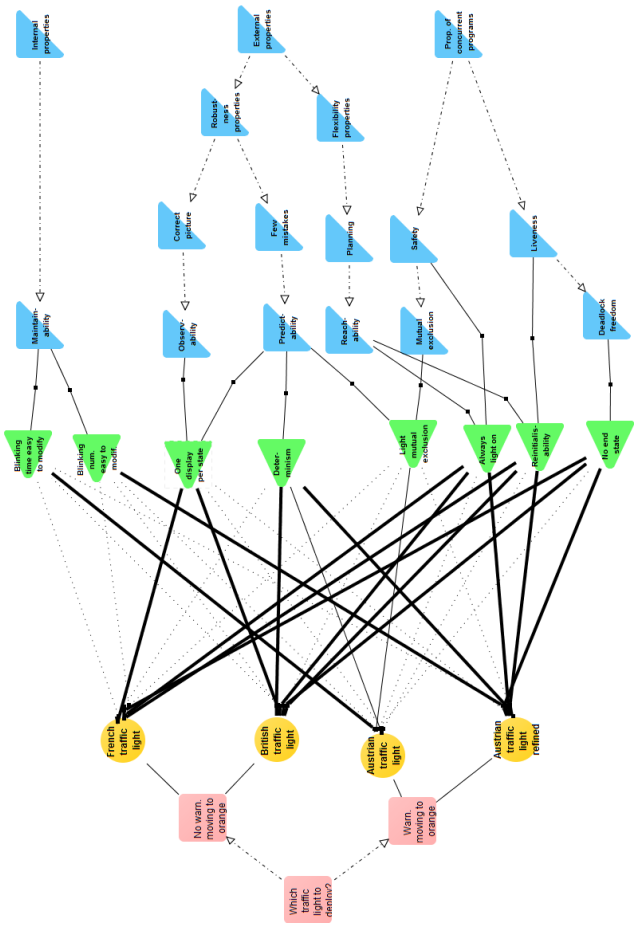


Fig. 12. DREAM diagram for the design options of the traffic light

The property “One display per state” is true (bold line) as or each state in the model; there is either a switching light on or a switching light off when entering the state.

4.4 Description of properties for the entire case study

Fig. 12 presents a summary of the properties that are true or false for the four behavioral model of traffic light presented above. It is interesting to note that the Austrian traffic light holds more properties than the other ones. This is because this traffic light

has more perceivable states (with different lights on and off) than the other ones and to the fact that the first two properties are only meaningful for Austrian traffic lights.

Accessibility is another property that can be related to the case study. Indeed, the fact that the lights are located on 3 different locations (top, middle and bottom) it allows color-blind people to understand the current status of the traffic light. Another design with 3 lights (of different colors) in the same slot would result in a traffic light with accessibility problems.

5 Discussions and Conclusion

This paper has presented a notation allows representing the hierarchical relationship between properties for computing systems in general but also adapted for interactive systems. This notation has been applied to exiting classifications of properties available in the literature of these domains.

We have used a set of behavioral models from a simple case study to connect an application to this hierarchy of properties. The notation can thus be used for comparing design alternatives as this has been demonstrated on the alternative traffic lights that are deployed in real life.

Further work will be devoted to a deeper understanding and representation of the various types of relationships that could connect two properties. For instance, the notion of inclusion could be represented as well as other more complex ones such as composition or inheritance.

6 References

1. Alan J. Dix: Abstract, Generic Models of Interactive Systems. BCS HCI 1988, 63-77 (1988).
2. Gram, C. and Cockton, G. (eds): Internal Properties: The Software Developer's Perspective. In Design Principles for Interactive Software, pp. 53-89, Springer US (1996).
3. Gram, C. and Cockton, G. (eds.): External Properties: the User's Perspective. In Design Principles for Interactive Software, pp. 25-51, Springer US (1996).
4. Hassenzahl M., Platz A., Burmester M., Lehner K. Hedonic and ergonomic quality aspects determine a software's appeal. CHI 2000: 201-208
5. International Standard Organization: "ISO 9241-11." Ergonomic requirements for office work with visual display terminals (VDT) – Part 11 Guidance on Usability (1996).
6. Lacaze X., Palanque P., Barboni E., Bastide R., Navarre D.: From DREAM to Realitiy: Specificities of Interactive Systems Development with respect to Rationale Management. In: Rationale Management in Software Engineering. Allen H. Dutoit, Raymond McCall, Ivan Mistrik, Barbara Paech (Eds.), Springer Verlag, Springer-Verlag/Computer Science Editorial, p. 155-172 (2006)
7. Lamport, L.: Proving the correctness of multiprocess programs. IEEE transactions on software engineering (2), 125-143 (1977).
8. Laprie, J. and Randell, B. 2004. Basic Concepts and Taxonomy of Dependable and Secure Computing. IEEE Trans. Dependable Secur. Comput. 1, 1 (Jan. 2004), 11-33

9. MacLean, A., Young, R. M., Bellotti, V. M. E. and Moran, T. P.: Questions, Options, and Criteria: Elements of Design Space Analysis. Lawrence Erlbaum Associates, 6, pp. 201-250 (1991).
10. Manna, Z., Pnueli, A.: A Hierarchy of Temporal Properties. ACM Symposium on Principles of Distributed Computing1990: 377-410 (1990).
11. Martinie, C., Palanque, P., Winckler, M., Conversy, S. DREAMER: a Design Rationale Environment for Argumentation, Modeling and Engineering Requirements. In proceedings of the 28th ACM International Conference on Design of Communication (SIGDOC'2010), September 26-29, 2010, São Carlos, Brazil. ACM Press. pp. 73-80.
12. Masip, L., Martinie, C., Winckler, M., Palanque, P., Granollers, T. and Oliva, M.: A design process for exhibiting design choices and trade-offs in (potentially) conflicting user interface guidelines. In: Proc. of the 4th international conference on Human-Centered Software Engineering (HCSE'12). Springer-Verlag, Berlin, Heidelberg, 53-71 (2012).
13. Palanque P. & Lacaze X. DREAM-TEAM: A Tool and a Notation Supporting Exploration of Options and Traceability of Choices for Safety Critical Interactive Systems. In Proceedings of INTERACT 2007, Lecture Notes in Computer Science 4662, p. 234-250 Springer Verlag.
14. Petrie H. and Kheir O.: The relationship between accessibility and usability of websites. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07). ACM, New York, NY, USA, 397-406 (2007).
15. Pirker, M. and Bernhaupt, R.: Measuring user experience in the living room: results from an ethnographically oriented field study indicating major evaluation factors. EuroITV 2011, 79-82 (2011).
16. Pnueli A.: Applications of Temporal Logic to the Specification and Verification of Reactive Systems: A Survey of Current Trends. LNCS n° 224 p.510-584. Springer Verlag (1986).
17. Pnueli, A.: The Temporal Logic of Programs. 18th IEEE symposium on the Foundations of Computer Science, 46-57 (1977)
18. Sasse M. A., Karat C.-M., and Maxion R.: Designing and evaluating usable security and privacy technology. In: Proceedings of the 5th Symposium on Usable Privacy and Security (SOUPS '09). ACM, New York, NY, USA, Article 16, 1 page (2009).
19. Section 508: The Road to Accessibility. Available at: http://www.section508.gov/
20. Sistla, A. P.: On characterization of safety and liveness properties in temporal logic. In: Proceedings of the fourth annual ACM symposium on Principles of distributed computing, pp. 39-48, ACM (1985).
21. Toulmin, S.E. (1958) The Uses of Argument. Cambridge: Cambridge University Press.
22. Vanderdonckt, J.: Development milestones towards a tool for working with guidelines. Interacting with Computers 12(2), 81-118 (1999).
23. Whitacre JM, Bender A.: Degeneracy: a design principle for achieving robustness and evolvability. Journal of Theoretical Biology 263(1), 143-153 (2010).
24. Wood W.A. Transition network grammars for natural language analysis. Communications of the ACM 13, 10 (October 1970), 591-606
25. Yan J. and El Ahmad A. S.: Usability of CAPTCHAs or usability issues in CAPTCHA design. In Proceedings of the 4th symposium on Usable privacy and security (SOUPS '08). ACM, New York, NY, USA, 44-52 (2008).