Modular Functional Descriptions

Bernhard Schätz
Technische Universität München, Fakultät für Informatik
Boltzmannstr. 3, D-85748 Garching bei München, Germany

E-mail: schaetz@in.tum.de

Abstract

The construction of reactive systems often requires the combination of different individual functionalities, thus leading to a complex overall behavior. To achieve an efficient construction of reliable systems, a structured approach to the definition of the behavior is needed. Here, functional modularization supports a separation of the overall functionality into individual functions as well as their combination to construct the intended behavior, by using functional modules as basic paradigm together with conjunctive and disjunctive modular composition.

1 Introduction

In many application domains reactive systems are becoming increasingly complex to cope with the technical possibilities and requested functionalities. The behavior provided by the system often is a combination of different functions integrated in an overall functionality; e.g., an embedded controller managing the movement of a car power window combines control of the basic movement, position control to restrict motor overload, as well as power management to avoid battery wear.

Implementing those combinations of individual functions is a complex and error-prone task. Since these functions in general influence each other, a non-modular development process complicates ensuring that the combined functionality ensures that the restrictions imposed by each individual function are respected.

Here, the use of functional modules can improve the development process by supporting the modular definition of the basic functions as well as their combination into the overall functionality.

1.1 Contributions

Modular functional development aims at supporting the development process of multi-functional reactive systems by use of *modular composition of functions*. To that end, we use

- *functions* as the modular units of system construction, which provide a data flow interface describing the values observed and controlled by a function as well as a control interface describing the activation and deactivation of functions.
- *disjunctive* and *conjunctive composition* as a means of combination, which allow to either alternatively or simultaneously combine functional behavior.

1.2 Related Approaches

Functions are modules of behavior, used for the construction of complex behavior from basic functionality. They offer interfaces for both data and control flow in a similar fashion introduced in [1][1]. They are similar to the modules introduced in [2], however allowing a more generalized form of (conjunctive) composition. [3] gives a more detailed description and a formalization of functions as used in the following.

2 Functional Modules

Functions form the building block of the approach presented here. Basically, functions are capsules of behavior, defined by their (external) interface in terms of data and control flow as well as their (internal) implementation. The data flow between the function and its environment is described in form of data signals exchanged between them, allowing the function to observe and control shared signals.

[1] [1] defines the data interface via ports, the control interface via connectors.

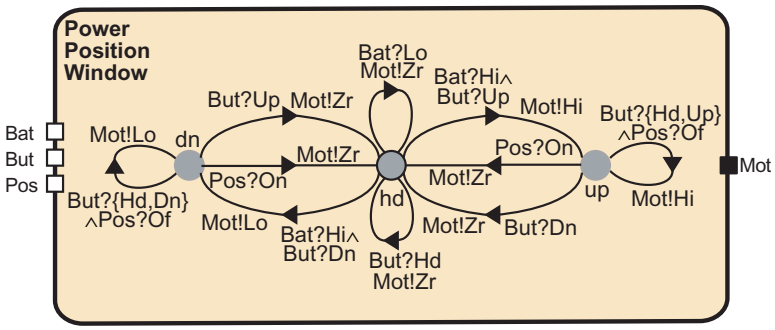

Figure 1. Refactored Power Position Window Function

The control flow between the function and its environment is described in form of control locations used to pass control between them, allowing the function to bet activated and deactivated.

Figure 1 shows a function PowerPositionWindow describing the functionality of a basic power window controller. The function observes user interactions via the But signal (with Up, Hd, and Dn signaling the up, hold, and down position of the switch), the current battery status via the Bat signal (with Hi and Lo signaling high and low voltage), and the current position of the window via the Pos signal (with On and Of signaling intermediate or end position of the window). It furthermore controls the motor of the window via the Mot signal (with Hi, Lo, and Zr signaling an upward, downward, or halted movement). The function can be entered and exited via the control location hd.

As suggested by the interface-locations, the behavior of the function is described in a state-transition manner. As shown in Figure 1, its internal control flow is described via locations dn, hd, and up (with hd being the interface location), as well as transitions between these locations. Transitions are influenced by observed signals and influence controlled signals. Thus, if control resides in location hd, value Hi is received via the Bat signal (Bat?Hi), and a value Up is received via the But signal (But?Up), then value Hi is sent via the Mot signal (Mot!Hi) and control is transferred to location up.

A transition can be understood as the most basic form of a function. Its interface is defined by the observed and controlled signals as well as by its start and end locations.

3 Decomposing Functionality

The functionality of PowerPositionWindow shown in Figure 1 combines three different functions:

1. Basic Window movement control, moving the window as requested by the interactions of the user,
2. Position Control, halting the window if reaching an end position,
3. Power Control, disabling window movement if lacking sufficient voltage.

Obviously, all three functions control the motor movement via the Mot signal, interacting to realize the overall behavior. However, their integrated behavior does not support a modularization of the behavior of the controller, failing to reflect the separation of concerns into individual functions. As a result, identifying these three functions and ensuring that the overall behavior implements these functions is complicated.

Figures 2, 3, and 4 show the corresponding basic functionalities. In contrast to Basic Window, functions Position Control and Power Control are decomposed into their sub-functions.

Position Control consists of Position Check, ensuring that the motor is restricted to intermediate positions, and Position Override, ensuring that the motor is stopped if an end-position is reached. As shown in the left-hand side of Figure 3, the sub-functions are disjunctively composed to Position Control, linked via their interface locations on and off, allowing to abort a controlled movement via the override function.

Similarly, Power Control consists of Power Check, ensuring that starting the motor movement requires sufficient voltage, and Power Override, ensuring that the motor is not activated in case of insufficient voltage. As shown in the left-hand side of Figure 4, again the sub-functions are disjunctively composed to Power Control via their interface location on, allowing to block a controlled movement via the override function.

4 Composing Functionality

To obtain the overall behavior of the controller of the power window, the functions introduced in Section 3 are

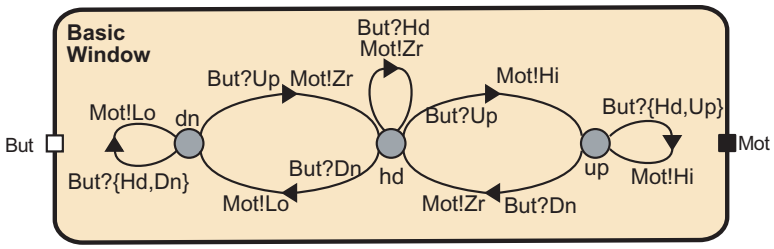

Figure 2. Basic Window Function

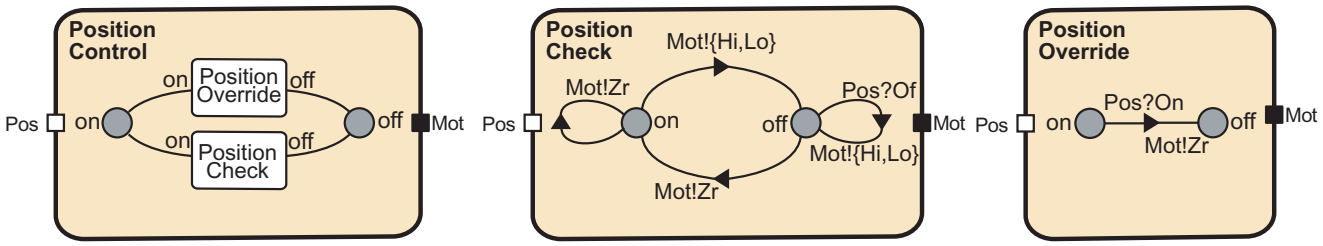

Figure 3. Position Control Function

combioned. Obviously, simple conjunctive composition does not lead to a reasonable behavior, since the above three functions are in conflict to each other. Therefore, a more sophisticated form of combination is needed, describing the priorities between these (sub-)functions.

Figures 5 and 6 describe these priorities composition. At the top level, the Power Position Window is realized by disjunctive combination of the Power Override function together with the Controled Position Window, ensuring that a lack of voltage may result in a blocked window movement. As Controled Position Window is obtained by conjunctive composition of Power Check and Position Window, any window movement is only initialized in case of sufficient voltage.

A similar construction is applied to ensure position control. As Position Window is realized by disjunctive composition of Position Override and Controled Window, detection of an end position may stop the movement of the window. By conjunctive composition of Position Check and Basic Window to form Controled Window, the basic window movement is restricted to intermediate positions of the window.

5 Conclusion

The increasing complexity of reactive behavior integrating different interacting functionalities requires a construction process supporting the modular description of individual functions as well as their composition into the overall behavior.

Therefore, we suggest functional modular development using functions as construction units, with transitions as the most basic form, as well as disjunctive and conjunctive composition to combine modules. Offering separation of concern by modular composition of functions, reasoning about the overall behavior is simplified by conjunctive and disjunctive construction of functionalities. Finally, using refactoring techniques for the simplification of the modular description, more constructive forms on the overall behavior can be obtained.

References

[1] Huber, F., Schätz, B., Einert, G.: Consistent Graphical Specification of Distributed Systems. In: Industrial Applications and Strengthened Foundations of Formal Methods (FME'97). LNCS 1313, Springer Verlag (1997) 122–141

[2] Henzinger, T.A.: Masaccio: A Formal Model for Embedded Components. In: Proceeding of the First International IFIP Conference of Theoretical Computer Science, Springer (2000) 549–563 LNCS 1872.

[3] Schätz, B.: Building Components from Functions. In: Proceedings 2nd International Workshop on Formal Aspects fo Component Software (FACS'05), Macao (2005)

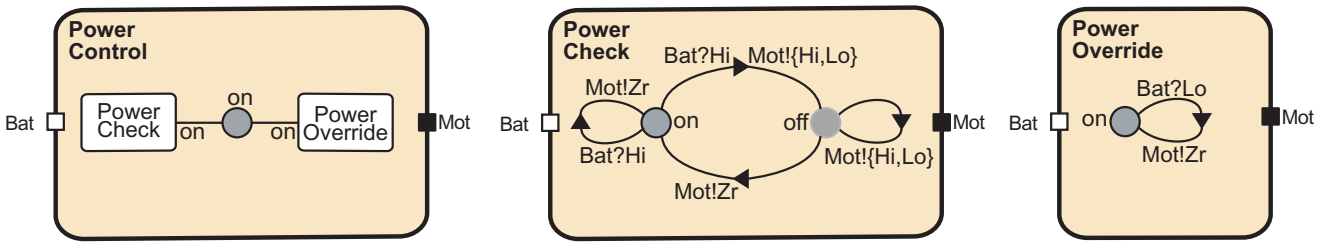

Figure 4. Power Control Function

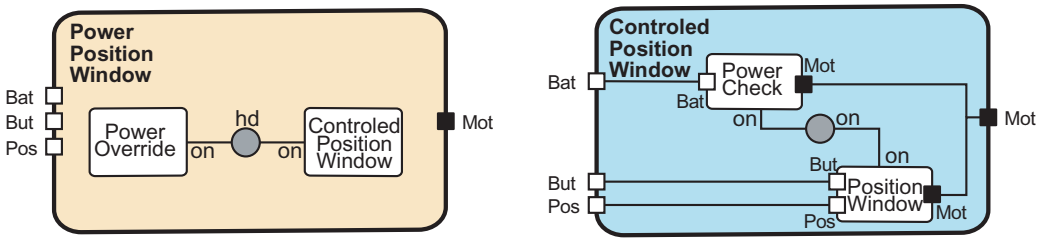

Figure 5. Power Position Window and Controled Position Window Functions

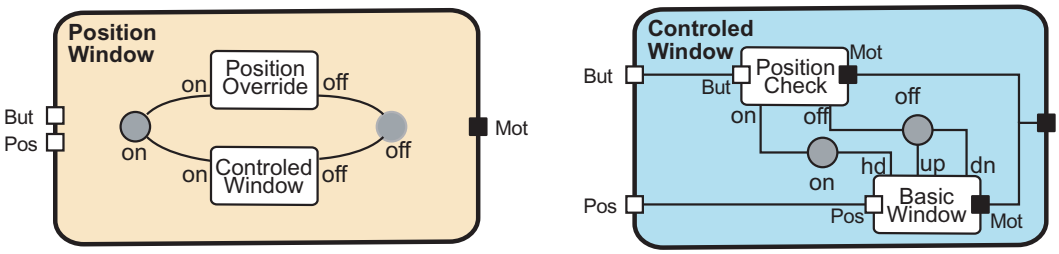

Figure 6. Position Window and Controled Window Functions