Part I

FRAMING THE PROBLEM AND CURRENT STATE-OF-THE-ART

You can put some informational part preamble text here. Illo principalmente su nos. Non message *occidental* anglo-romanic da. Debitas effortio simplificate sia se, auxiliar summarios da que, se avantiate publicationes via. Pan in terra summarios, capital interlingua se que. Al via multo esser specimen, campo responder que da. Le usate medical addresses pro, europa origine sanctificate nos se.

Part II

DESIGN ITERATIONS AND CONSTRUCTING A THEORY

You can put some informational part preamble text here. Illo principalmente su nos. Non message *occidental* anglo-romanic da. Debitas effortio simplificate sia se, auxiliar summarios da que, se avantiate publicationes via. Pan in terra summarios, capital interlingua se que. Al via multo esser specimen, campo responder que da. Le usate medical addresses pro, europa origine sanctificate nos se.

6 SEMANTIC CONNECTIONS THEORY

Essentially, all models are wrong, but some are useful.

— George Box

Extracting from the lessons learned during the three design iterations, a theory was developed for interacting with a system of devices in a ubiquitous computing environment. This chapter introduces a theory of semantic connections, in which the connections and associations between devices play a central role. Semantic connections focus on the semantics—or meaning—of the connections between entities in a smart environment.

The theory may be used to analyse (i.e. understand, explain and predict) what happens with interaction events and other interaction data when devices are interconnected and form an ecology of smart objects. As an introduction to the Semantic Connections theory, we first focus on the Semantic Connections user interaction model.

6.1 USER INTERACTION MODEL

A user interaction model for semantic connections is shown in Figure 31. It describes the various concepts that are involved in the interaction in a smart environment and shows how these concepts work together. The interaction model was inspired by the Model-Control-RepP-RepD (MCRpd) by Ullmer and Ishii [103] which in turn was based on the Model-View-Controller (MVC) model, both of which are described in Section 3.

We first distinguish between the physical and digital domains of the user interaction. A user does not observe directly what is happening in the digital domain, but experiences the effect it has in the physical world by interacting with various smart objects. Semantic connections exist between these objects. By interacting with the objects, users create a mental model of the system that they are interacting with, which only partly includes the digital domain. The digital part manifests itself in the physical world as data, media and services.

The terminology of inherent, augmented and functional feedforward and feedback is adopted from [117] and was previously introduced in Section 2.5.

When a user interacts with a smart object, he/she senses feedback and feedforward, directly from and inherent to the controls of the device (inherent feedback), digital information augmented onto the physical world (augmented feedback) and perceives the functional effect of the interactions (functional feedback). The user actions in the physical world are transformed into interaction events and device state changes. This interaction data in terms of user intentions

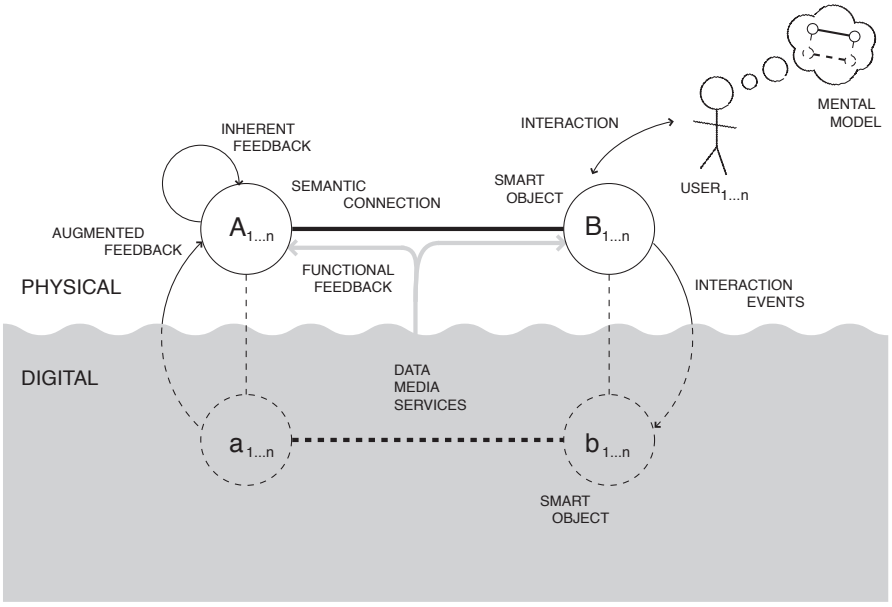

Figure 31: Semantic connections user interaction model

is stored in the smart space. The notion of a smart space means that data are stored either by an information broker or the smart objects themselves, and can be accessed by the other smart objects in the smart space. We use the term smart environment as a broader term to refer to both the digital and physical spaces, which include both the smart space and the smart objects.

Note that in Figure 31 the arrow on the left shows the feedback (or output) of the smart object, and on the right it shows the input (the interaction event). This to avoid repetition, as they may and in most cases will, occur on both sides.

6.2 SMART OBJECTS

Smart objects are the devices that are connected to the smart space, enabling them to share information with one another. We now define a smart object as follows:

SMART OBJECT A smart object is a device with both computational and network communication capabilities that can be uniquely identified in both physical and digital space.

According to our definition, an Near Field Communication (NFC) enabled smart phone is a smart object. A WiFi-connected lamp is also a smart object, given that it can be physically identified, for example by proximity based on signal strength or Radio Frequency Identification (RFID).

Note that our definition does not define where the Knowledge Processor (KP) is situated. For simple smart objects, such as a smart light bulb, the actual KP that is communicating to the information broker may be a virtual entity running on any device in the network.

In terms of our definition, a light switch with an RFID tag is not a smart object. A software agent running on a Graphical User Interface (GUI) (e.g. Microsoft Office's Clippy[1]), is not considered a smart object, even though it is visually perceivable. Despite its apparent

1 http://en.wikipedia.org/wiki/Office_Assistant

physical existence, i.e. physically and digitally identifiable, it is mediated by a computer. In such cases the computer is considered the smart object and not the agent running on it, as the agent is not primarily a physical entity. In the following subsections we describe the concepts specific to the smart objects themselves.

6.2.1 *Identification*

For semantic connections to work in the way they are envisioned in this thesis, a smart object needs to be uniquely identifiable in both the physical and the digital domain. In the physical space it needs to be both user-identifiable and machine-identifiable. A device that is tagged with an RFID tag is machine-identifiable in the physical space, and the unique identifier read from this tag is also linked to the digital representation of the smart object. NFC—using a near field channel like RFID or infrared communication—is an interesting case, because it allows for direct manipulation of wireless network connections by means of proximal interactions [88].

Of course, there are many ways in which a smart object may be identified. An IP address makes a device easily identifiable in the digital space, but it is difficult to create a physical representation of this identity. Consider the case where IP addresses are printed on stickers and stuck on computers to make them easier identifiable to IT service personnel. One solution to making these stickers machine-identifiable again is to use **QR!** (QR!) codes, two-dimensional barcodes which can be identified by mobile phone cameras.

6.2.2 *Interaction primitives*

We defined interaction primitives as a way to describe the user interaction capabilities of smart objects in ubiquitous computing environments. These interaction primitives are based on the work of Foley, Card and others introduced in Section 2.4.

Interaction primitives were introduced in Section 4.2.2.

The key on a keyboard labelled "A" is an interaction primitive, as pressing it not just changes the key's `Up` state into a `Down` state, but carries the meaning to *produce* a character "A". A gesture `SwipeLeft` on a touchpad is also an interaction primitive, as this is the smallest addressable element to still have meaning. Describing the input on a lower level would cause it to lose its meaningful relation to the interaction. A touchpad itself is not an interaction primitive but rather an input device. An interaction with the touchpad, annotated with its meaning, can be an interaction primitive. A GUI is not an interaction primitive, but a GUI element can be.

Interaction primitives are described in terms of their physical properties that are meaningful to a user. For example, an unlabelled button should not only be represented in terms of its `On` and `Off` states, but

also whether it is in a `Up` or `Down` state. This enables the mapping of physical, generic interaction primitives like a rocker switch to specific high-level events like `VolumeUpEvent`.

An interaction primitive also has a range measure that describes the range of possible values that it can take on. This makes it easier to determine if and how they can be mapped to specific interaction events.

Interaction primitives and interaction events together form an *interaction path* [34]. As an example, a typical interaction path would be:

$$\texttt{VolumeSliderLeft} \rightarrow \texttt{SlideLeftEvent} \rightarrow \texttt{VolumeDownEvent}$$

where the `VolumeSliderLeft` is an interaction primitive mapped to the `SlideLeftEvent` interaction event. Based on the available context information, this can in turn be mapped to a more specific `VolumeDownEvent`.

When modelling interaction primitives, only that which is meaningful to be shared with other devices is considered. It is not necessary to describe interactions that are internal to the device and that are not shared. An accelerometer, for example, may be modelled as a separate device, sharing the raw accelerometer data to be used by other devices. However, when integrated into a smart phone, the accelerometer's data can often be abstracted as part of an interaction path, e.g. to only share the orientation of the device, or specific gestures measured with the accelerometer. In this case, the raw values may only need to be available locally on the device, to be used by the developers of other device-specific applications.

What can be inferred based on these descriptions of interaction primitives? We could, for example, infer input devices types based on their properties:

- If an interaction primitive measures position in one dimension, we can infer that it is a type of `Slider`.
- If a phone has a `Slider` and a speaker has `Volume` functionality we can infer that these devices can be connected, where the one device is used to control the volume of the other device.
- By describing a computer mouse in terms of its manipulation operators and possible states (measuring movement on x,y-axis and states Up,Down), we can infer it to be a `Slider`, `2DTablet` or a `Button`.

One of our academic partners in the Smart Objects For Intelligent Applications (SOFIA) project, the University of Bologna, created an independent implementation of our interaction primitives, which was published in the Springer Journal of Personal and Ubiquitous Computing [10].

Figure 32: Nokia Play 360° speaker system and N9 mobile phone

6.3 SEMANTIC CONNECTIONS

Semantic connections is a term we introduced [109, 110] to describe meaningful connections and relationships between entities in an ecosystem of interconnected and interoperating smart objects.

Semantic connections were introduced in Chapter 3.

The connection between a remote control and a wirelessly controllable (on/of or dimmable) light bulb is a semantic connection. The connection exists between two smart objects that can be physically identified and connected through physical proximity. The connection's communication technology is unknown to its user and the remote control and light are conceptually linked by users, based on the perceived behaviour.

Another example of a meaningful connection is the Nokia Play 360° speaker system, shown in Figure 32, where music can be streamed wirelessly to the speaker using an NFC-enabled smart phone. By touching the phone to the top of the speaker, a connection is created that conceptually "carries" the music from the phone to the speaker.

The WiFi connection between a smart phone and a WiFi router is not a semantic connection, as the connection in itself has no clear meaning. A USB cable by itself is also not considered a semantic connection.

Semantic connections make up a structural layer of inter-entity relationships on top of the network architecture. These connections can be the real, physical connections (e.g. wired or wireless connections that exist between devices), or conceptual connections that seem to be there from a user's perspective. Semantic connections exist in both the physical world and the digital domain. They have informative properties which are perceivable in the physical world. However, some of these physical qualities might be hidden by default, and only become visible on demand by means of a mediating interaction device. The digital parts of semantic connections are modelled in an ontology. There may be very direct mappings, e.g. a connection between two real-world entities may be modelled by a `connectedTo` relationship between the representations of these entities in an ontology. Sometimes the mapping is not so direct, for example where a semantic transformer is used. Semantic connections have several properties, which are explained in the following subsections.

6.3.1 *Directionality*

As discussed in Section 5.2, we consider a semantic connection to have a specified direction, or to be bidirectional/symmetric. Smart objects that are connected should then be identified as sources and/or sinks. Directionality may intentionally be specified through user action, or it can emerge from the capabilities of the smart objects e.g. connecting a source to a sink will automatically create a connection going from the source to the sink.

6.3.2 *Transitivity*

When connections have directionality and multiple devices (i.e. a minimum of three devices) are involved, devices can also act as bridges, transferring data due to transitivity. For example, if a music player is connected to speaker A, and speaker A is connected to speaker B, speaker A acts as a bridge between the music player and speaker B.

6.3.3 *Permanent and temporary connections*

Semantic connections can vary in persistence. Connections can be made during an interaction cycle involving several devices to transfer content or data from the one device to another, and the connection then stops existing when the interaction cycle is completed. Connections can also be used to configure more permanent information exchange between entities in a smart space, much like setting up a connection to a wireless network router. These permanent connections will persist, and will be automatically reconnected every time the smart objects that are connected co-exist in the same smart space.

6.3.4 *Connections connect different entities*

Connections can exist between smart objects, people and places. Not only objects and devices have meaning in a system of networked devices — according to [84] physical location within the home and device ownership (or usage) are of central importance for understanding and describing home networks by users. Ownership can be seen as a connection between a device and a person. Connections from and to places or locations can be seen as a way of structuring contextual information such as location. With very personal devices (such as smart phones and laptops or tablets) we can, when these devices are used in an interaction, implicitly infer the user's identity. With shared devices, we need a way to identify the user. In such cases, making explicit connections from the device at hand to something personal of the user (e.g. a phone or keychain) may be a way to indicate identity.

6.4 SEMANTIC TRANSFORMERS

Semantic transformers were introduced in Section 4.2.1.

Semantic transformers were first defined in [74] as virtual entities that transform one type of information into another when a direct mapping is not possible. They transform user actions into interaction events and perform matching and transformation of shared data and content. Semantic transformers enable interoperability between devices by utilising device capability descriptions and content types to determine how devices may interoperate.

Semantic transformers can be used to map and transformed shared content between smart devices, for example a service that transforms a music stream into coloured lighting patterns that can be rendered by a lighting device. Semantic transformers can also be used to transform physical actions (such as pressing a button or performing a gesture) into representational events like adjusting the level of lighting in a room, or the adjusting the volume of a speaker. Semantic transformers may also be employed to perform simpler transformations such as inverting values.

Physical identifiable objects are not considered semantic transformers and should rather be modelled as smart objects. Semantic transformers are services, and therefore have no physical appearance or tangible form. They can only be perceived through the smart objects they transform the information for. A semantic transformer is not considered a smart object, as it is a virtual object and not addressable in the physical environment.

6.5 FINITE STATE MACHINE EXAMPLE

We now use Finite State Machines (FSMs) to model and explain the different concepts introduced so far. FSMs allow us to talk about user

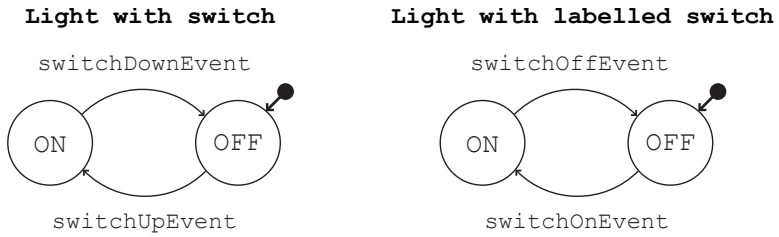

Figure 33: FSMs for a simple light with a switch and a light with a labelled switch

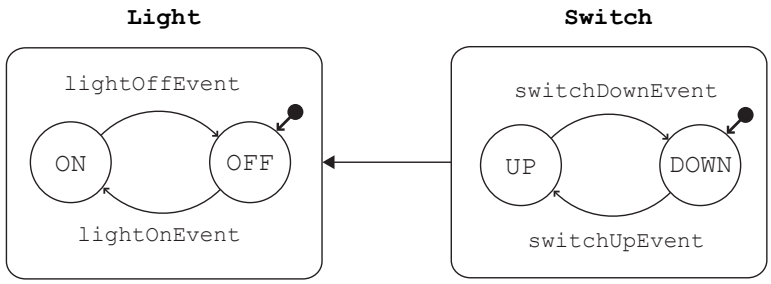

Figure 34: Light and light switch as two separate smart objects with a semantic connection

interaction in a way that describes how users could think about user interaction, but that still makes sense to interaction programmers and designers. Turing machines and other sophisticated models of computing do not describe that what users think about [101]. The use of FSMs also encourages simplicity.

As a first example, consider a simple light with an up/down switch as a single device (seen in Figure 33 on the left). There are two states (`On/Off`), an initial state (`Off`) and two events (`SwitchDownEvent/SwitchUpEvent`) that cause transitions between the states. If the switch is labeled, we can use more specific (meaningful) wording, for example `switchOffEvent` instead of `switchDownEvent` (as shown in Figure 33 on the right).

In Figure 34 one of the simplest examples of a semantic connection is shown - a light (as a smart object) connected to a simple up/-down switch (a second smart object). The light consists of two states (`On/Off`) with an initial (default) state of `Off`, and two events (`LightOnEvent / LightOffEvent`) indicating the transitions between these states. Boxes with rounded corners are used to signify smart objects, while the semantic connection is indicated using a solid arrow point. Using arrows to denote semantic connections allows us to specify a direction for the connection. Note that the light has functional feedback, with perceivable light when it is switched on. The switch on

The concept of directionality was described in Section 6.3.1.

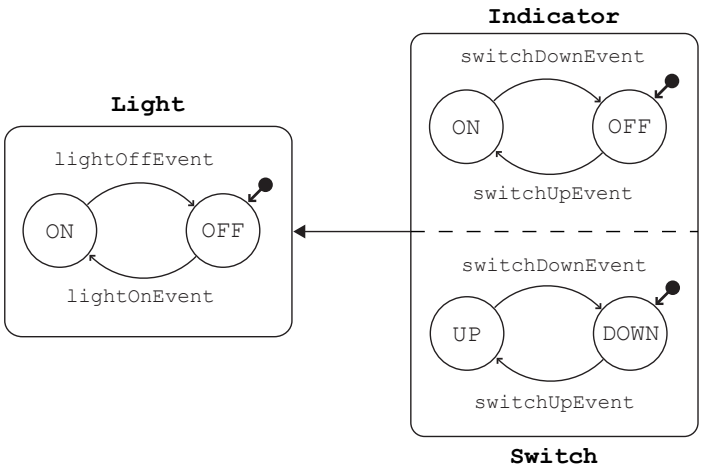

Figure 35: Light connected to light switch with augmented feedback

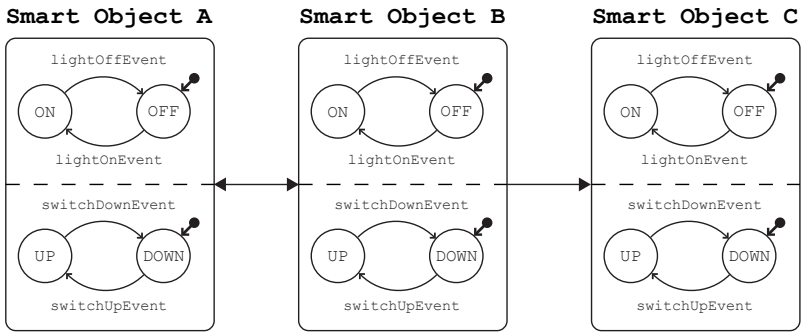

Figure 36: FSM showing semantic connection with symmetry

the other hand has inherent feedback, with a perceivable `Up` or `Down` state.

We can create mappings between the events to create an interaction path (see section 6.2.2), for example we use

$$\text{SwitchUpEvent} \rightarrow \text{LightOnEvent}$$

to indicate the most meaningful *default* mapping. It should of course be possible to change this mapping, for example by using a semantic transformer that inverts mappings between devices.

In the case where a smart object is not in the same physical location as the smart object it is connected to, additional augmented feedback may be required. Consider the case where the light switch may be in a different room than the light - we could use an indicator on the switch to give augmented feedback to show whether the light actually switched on. This is shown in Figure 35.

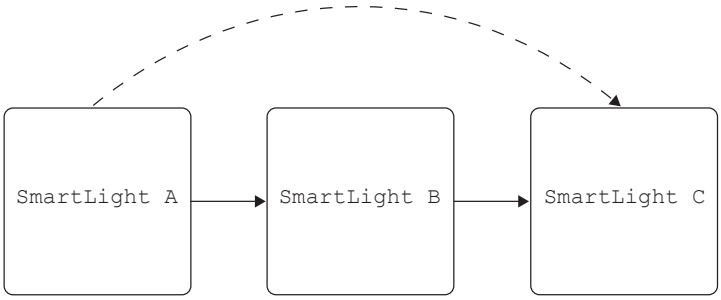

Figure 37: FSM showing a semantic connection with transitivity

A more complex example is shown in Figure 36. In this example there is a symmetric (bidirectional) connection between smart object A and B, with the result that pressing the switch on smart object A will turn the light in smart object B either on or off, and vice versa, B will control A. Since B is connected to C, actions on A and B, will also be reflected on C. On the other hand, pressing the switch of C will have no effect on either A or B. Due to the symmetric connection, we expect A and B to be in an identical state. How exactly this is implemented is up to the designer.

One possible solution for such a light switch is a push button with an indicator light, such that the switch able to change its state by itself.

In Figure 37 we use the SmartLight abstraction to denote the FSM of a smart light as shown in previous figures. When SmartLight A is connected to SmartLight B, and in turn is connected to C, transitivity allows us to infer a direct connection (indicated by a dashed arrow) between A and C. Pressing the light switch on A will in this case affect both B and C.

We use locked/unlocked icons next to semantic connections to indicate persistence (see Figure 38). The locked icons between A, B and D indicate persistent connections between those objects, and a persistent transitive connection is then inferred between A and D. This means that if smart object D moves to another location, all three the connections (including the A→D connection) continue to exist. The connection between B and C is temporary, which means that the inferred transitive connection between A and C is also temporary. If smart object C moves to another location, both the B→C and A→C connections will be removed.

In Figure 39 we show semantic connections between smart objects and specific locations, where the dashed-double circle denotes a location. This places semantic connections between places and objects on the same abstraction level. We use semantic connections between smart objects and places as a way to structure relevant contextual information. In our example in Figure 39 we cannot infer that a user actually is able to observe the functional feedback of switching the light on and off, as they are not located in the same space, and might not be able to see the light. The importance of feedback and feedfor-

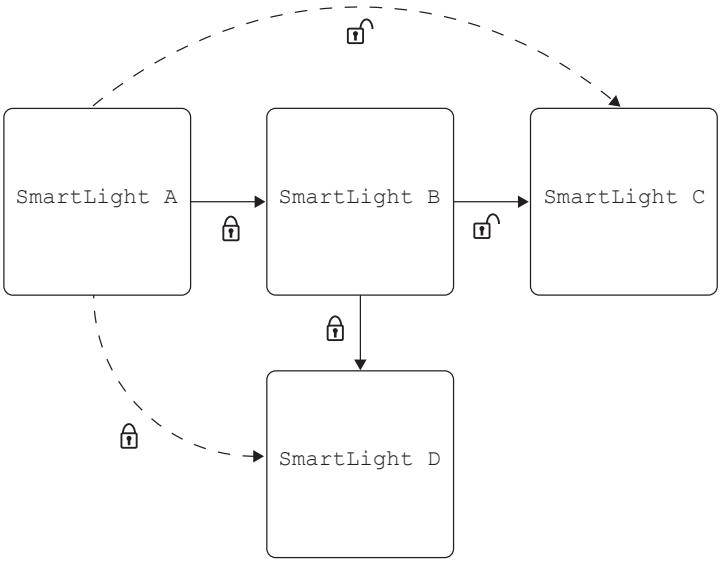

Figure 38: FSM showing a semantic connection with transitivity and persistence

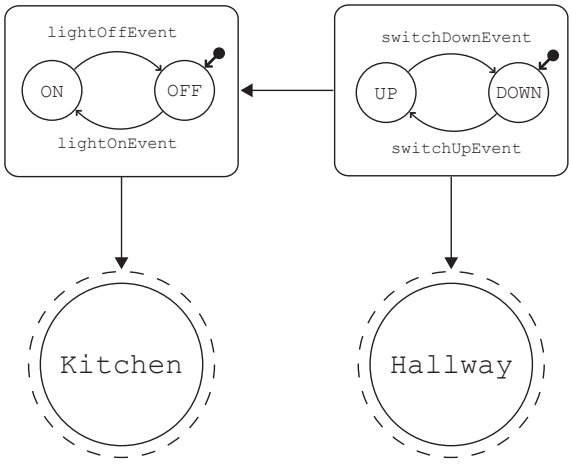

Figure 39: FSM showing semantic connections between smart objects and places

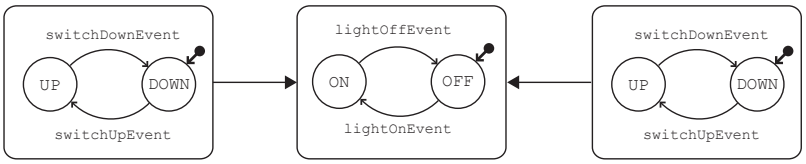

Figure 40: FSM showing a situation where priority is an issue

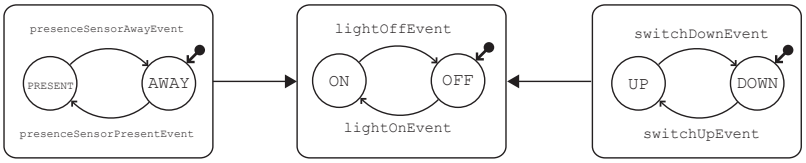

Figure 41: FSM showing incidental (presence sensor) and intentional (light switch) interactions

ward and how they should be handled between different locations is described in more detail in section 5.4.1.

When two switches are connected to the same light as is shown in Figure 40, the issue of priority arises. We define the most meaningful default to be that the last event that occurred has priority. In Figure 41 where the one interaction is incidental, generated by a presence sensor, and the other is intentional (as described in Section 2.5.1), the intentional interaction takes priority.

6.6 FEEDBACK AND FEEDFORWARD

If we view our concept of semantic connections in terms of the Interaction Frogger framework (as discussed in Section 2.5), the following interesting insights emerge.

6.6.1 *Feedback of objects*

When we consider multiple interconnected smart objects and the functionalities and services they provide, information like feedback and feedforward gets spatially distributed. A user may operate a device, receiving inherent feedback locally, but receiving augmented and/or functional feedback remotely.

As inherent feedback is inherent to the operational controls of the device, these reside only in the physical world and are local to the device. We thus do not model this feedback in the digital domain.

Augmented feedback is feedback that is augmented from the digital domain onto the physical world. This type of feedback is subject to change when devices are connected to other devices. In the domain of networked digital artefacts, functional feedback is of a digital nature. Data, media and services that exist in the digital domain become available in the physical world, through the various devices and their connections. In Figure 31, the several types of feedback are indicated.

Although many functionalities of digital devices can be regarded as displaying media, data or services, for some simple functionalities this seems problematic. If we, for example, look at functional lighting, it seems that the presence of light as the functionality of a lighting device is not a concept that is part of the digital domain. However, if we view a lighting device as a networked smart object, the presence of lighting, based on some sensor data, can be regarded as the functionality of a digital service.

Refer to Van der Vlist's thesis [108] for more detail on how the theory of product semantics can be applied to feedback and feedforward.

6.6.2 *Feedback of connections*

Inherent feedback becomes feedback that is mediated through an interaction device used to make or break the connection, as one can not manipulate a wireless connection directly. This inherent feedback may however be closely related to the action of making or breaking a physical connection, like a snap or click when the connection is made or broken. Augmented feedback to indicate a connection possibility or an existing connection may be in the form of lights, or in the form of projected or displayed lines. Functional feedback is information about the actual function of the connection, like music playback from a speaker that was just connected to a media player. This type of feedback always reaches the user through the devices being connected.

6.6.3 *Feedforward*

Inherent feedforward, conceptually similar to the notion of affordances [76], provides information about the action possibilities with the devices or the individual controls of an interface. Inherent feedforward is always physical and locally on the device. However, when devices or objects are part of a larger system, feedforward also emerges where interaction possibilities between objects exist (e.g. a key that fits a lock, a connector of one device or cable that fits another). The same holds for augmented feedforward, where lights, icons, symbols and labels provide additional information about the action possibilities. These may concern the action possibilities locally at the device, as well as action possibilities that concern the interaction with other devices in the environment.

While inherent and augmented information are primarily concerned with "the how", functional feedforward communicates "the what",

the general function of the device or the function of a control. This type of information often relies on association, metaphors and the sign function of products, and are described in theories such as product semantics and product language. With multifunctional devices, and even more with smart objects, this becomes increasingly difficult. Introducing the concept of semantic connections tries to address these problems, therefore the functional feedforward is the main challenge when designing semantic connections. Functional feedforward gives information about the function of the semantic connection before the interaction takes place. Properly designing functional feedforward is therefore the crucial part of understanding semantic connections, smart services and smart environments.

An example of where functional feedforward was used in the third design iteration is described in Section 5.4.1.

Wensveen [117] further proposes that in interaction, these types of information can link action and function together in time, location, direction, modality, dynamics and expression. Strengthening these couplings between action and function will lead to richer and more intuitive interactions [116].

We can also view semantic connections in the Frogger framework in more general terms. Although semantic connections are not a physical device or product, but rather describe the structure or configuration of a system of devices, the Frogger framework can teach us important lessons. When we look at the link between action and functional information in time or location, a strong link would mean they coincide in time and location. For location this would mean that the connection that is made between devices corresponds to the location of the actual devices in physical space. But also that the feedback that is provided is coupled to the action in time an location. Additionally, the direction of the action of connecting/disconnecting devices, being moving devices towards or away from each other, strengthens the coupling in terms of direction. Also, the direction of the action could have a link to the directionality of the semantic connection that is made (e.g. the order in which endpoints of a connections are defined). Couplings in dynamics (of the action) can be used in similar ways and may express the persistence of the connection that is made.

6.7 DISCUSSION & CONCLUSION

In this chapter we introduced our Semantic Connections theory and used finite state machines to model and explain the different concepts. We defined the following main concepts:

- Smart objects, and the means to describe them in terms of a unique physical and digital identity
- Interaction primitives, and how they can be used to describe the user interaction capabilities of smart objects

- Semantic connections, and how they can be used to model meaningful connections between smart objects
- Semantic transformers, and how they transform information from one type into another

We identified some of the principles of semantic connections, including directionality and transitivity, as well as permanent and temporary connections. We also identified principles of the various types of feedback and feedforward that are required, not only for connections but also for smart objects.

The importance of being able to uniquely identify smart objects in both the physical and digital space, as well as sharing their interaction capabilities and states, was shown, including how it was grounded in the theory of interaction models by Nielsen, Card and others.

We showed how augmented and functional feedback and feedforward can help users to better predict the functional result of the connections they create. Functional and augmented feedback also showed to be key in maintaining the causal links between user action and function, distributed over interconnected smart objects.

A fundamental difficulty encountered during the implementation of the feedback and feedforward (and which is also a big challenge in interoperability in general), is what we call the *awareness paradox*. To foster emergent functionality, efforts are aimed at enabling smart objects to interoperate without their combined functionality being specifically designed. This means that the smart objects are unaware of each other, exchanging information though an information broker. For the users however, it is imperative that smart objects show behaviour as to appear to be aware of each another.

The way out of the paradox is to make use of proper use of feedback and feedforward that can be generated at runtime. Since the connections that may be created during use are not known at design time, smart decisions have to be made on how to describe the interaction events and functionalities that are shared.

By describing feedback and feedforward of the semantic connections as a result of the match in capabilities and functionalities, and having the semantic transformers and sink objects (instead of the source) produce the preview and indicator feedback, we make sure that they are capable of displaying (i.e. in the widest sense of the word, not limited to the visual modality) this feedback. Our reasoning is that, if a sink can be the sink of a functionality, it should also be capable of giving feedforward and feedback for this functionality.

Examples of preview events were shown in Chapter 5.

Moreover we showed that semantic connections and using semantic transformers to create services is an appealing idea, leading to additional and, more importantly, more meaningful functionalities of ensembles of existing devices. This may reduce the number of devices needed in our daily lives by reducing redundant devices.

Our Semantic Connections theory provides a foundation for modelling user interactions with interoperating smart objects in smart environments, and therewith the possibility to improve the interoperability among them. We considered the notions of feedback and feedforward to enhance perception of connectivity and the perceived causality between user action and feedback.

In the next part of the thesis, we will look at other concepts and techniques that can be used in smart environments, like device capability modelling and event modelling. Similarly to how we extracted the theory of semantic connections from the work completed in the design iterations, these more general concept and techniques are also based on the work done during the three design iterations.

Part III

CONTRIBUTIONS AND EVALUATION

In this part of the thesis, the more general concepts and techniques that can be applied to ubiquitous computing are described. These concepts and techniques were extracted from work done during the three design iterations.

Part IV

APPENDIX