Supernatural Superserious: Reactive Executable Markers

Lukas Eichelberger
lukas.eichelberger@student.unisg.ch
Interactions Research Group,
University of St.Gallen
St. Gallen, Switzerland

Simon Mayer
simon.mayer@unisg.ch
Interactions Research Group,
University of St.Gallen
St. Gallen, Switzerland

Ganesh Ramanathan
ganesh.ramanathan@student.unisg.ch
Interactions Research Group,
University of St.Gallen
St. Gallen, Switzerland

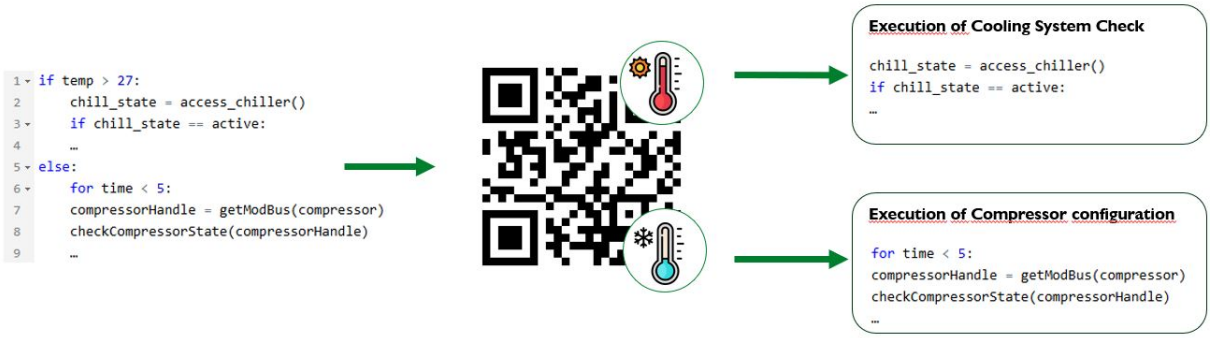


Figure 1: Illustration of the Reactive Executable Marker concept

Abstract

Machine-readable markers, such as QR codes, are used in a wide range of applications. These markers usually feature a single static payload that directs users to further application-relevant information, e.g., on a website or in local data resources. Recent research has introduced advanced versions of such markers: *reactive* QR codes that carry multiple payloads that are revealed depending on the marker's context and *executable* QR codes that carry executable code as payload. Building on these works, we introduce *Reactive Executable Markers* (REM). Based on exemplary application scenarios, we illustrate how REMs enable the offloading of the interpretation of environmental information to the REM designer without relying on network connectivity. Finally, we present a REM demonstrator and discuss the potential future evolution of REMs.

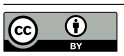
CCS Concepts

• **Human-centered computing** → **Interactive systems and tools**.

Keywords

Thermoresponsive, Thermochromatic, Interactive QR, Executable QR

ACM Reference Format:
Lukas Eichelberger, Simon Mayer, and Ganesh Ramanathan. 2025. Supernatural Superserious: Reactive Executable Markers. In *Companion of the the 2025 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp Companion '25), October 12–16, 2025, Espoo, Finland.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3714394.3754409



UbiComp Companion '25, Espoo, Finland

ACM ISBN 979-8-4007-1477-1/2025/10
https://doi.org/10.1145/3714394.3754409

1 Motivation

Machine-readable markers such as barcodes and Quick Response (QR) codes have gained widespread adoption and their applications today range from product identification in supermarkets to logistics and payments. To enable these applications, the markers usually encode a single static payload that directs users to a resource that may be accessed for further information about the scanned object or process, or that may allow users to interact with the object.

Recent research has introduced several fascinating advancements of such markers. On the one hand, Jenss and Mayer [5] demonstrated how *thermoresponsive ink* may be used to create QR codes that react to environment conditions and that, as a side effect, become *interactive*—that is, they display a different code depending on user interaction before scanning. On the other hand, Scanzio, Cena and Valenzano [8] demonstrated how QR codes may embed program code that is executable directly, that is, without for instance requiring to de-reference the QR code to a URL and accessing this URL.

In this paper, we introduce, discuss, and demonstrate a type of machine-readable marker that combines the concepts introduced in these publications to create markers that are both reactive to their context and directly executable. We propose that this combination is not only conceptually fascinating per se—as it holds the potential to *directly and immediately* embed physical parameters into program code through thermo-, hydro-, baro-, or chemoresponsive interfaces—but that it also has direct application in the Internet of Things (IoT) and embedded sensing fields.

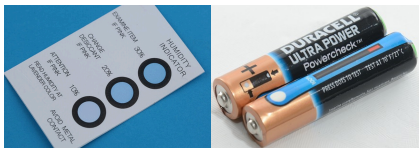


Figure 2: Functional inks such as the label on the left with ink that is reactive to humidity, or one on the battery on the right that is reactive to electrical potential, has been used for long in the industry. Instead of manual and off-band transfer of instructions, such as in these examples, our proposed REMs embed programs that are directly actionable by the automation system. Images from www.benz-packaging.com and www.duracell.com

2 Related Work

1D barcodes were the first widely used visual machine-readable markers. They were introduced in the second half of the twentieth century with the captivating idea that visually encoding information (e.g., a product identity) so that a machine can automatically decode it at a later point in time could (and did) transform logistics. During this process of industrial adoption, a plethora of (specialized) derivatives were introduced. One example is QR codes that overcome some problems with visual occlusions and blur through built-in error correction while bringing the potential to boost the amount of transmitted data (to almost 3 kilobytes), which can be further increased through color construct codes; other specializations include the GS1 Data Bar for marking particularly small products [1, 2].

The development of visual machine-readable markers continues to this day: most relevant for our proposed system is a recent paper by Scanzio, Cena and Valenzano who recently introduced **executable** QR (eQR) codes [8]. While typical QR codes usually encode information that directs users to a resource stored on a remote server (e.g., a Website) or to another local or remote resource (e.g., a mobile app), eQR codes *directly embed program code* that can be executed in a virtual machine on the reader device. Therefore, all application-specific information can be encoded in the QR-code and code execution can occur without dependence on network connection or local data availability. The authors have furthermore proposed a scripting language for this purpose, QRscript, and discuss applications of eQR codes in the IoT space and beyond in [10].

Functional inks have been used in the industry for the past few decades as medium to convey the past and current state of the environment in which an artifact was or is situated, and therefore how the artifacts should be further handled. Figure 2 shows two such examples. Inks have been developed that react to diverse physical, chemical, and biological states of the environment. Examples include inks that change color based on temperature, humidity, pressure, pH (Litmus being a well known example), light, and biological pathogens [4, 6, 7, 11]. Such functional inks, apart from being sensing elements, can also function as active electrical components that change their color depending on the amount of current flowing through it. For example, an ink that is sensitive to applied voltage is used to indicate the approximate state of a battery (See Figure 2). However, the interpretation of the state indicated by the inks has hitherto been considered only for manual reasoning by humans. Towards making device state that is exposed through functional inks machine-readable, a patent by Guinard et al. [3] proposed 2D barcodes that can alter the exposed information of a tag—i.e., the readable data—based on the tag's current environment (e.g., thermal, bariometric, or chemical) by using functional inks. Building on this concept, Jenss and Mayer [5] proposed the QRUco system as an approach to create cheap reactive visual machine-readable markers that are compatible with the QR standards; technically, QRUco uses functional ink to embed three secondary markers into the finder patterns of a standard QR code. Regarding application, [5] proposes that users may interact with these markers through warmth generated by rubbing (for thermochromic ink), pressing (for barochromic ink), etc.—all these interactions vary the appearance of the marker, and hence the embedded information, while leaving the primary QR code intact. Though approaches such as [5] make the state machine-readable, the interpretation in this case still has to rely on a service that is reachable over the network or custom software implementation on the reader. We argue that not only the interpretation, but the recommended action *itself* can be encoded by the designer of the artifacts—may that be of fruit trays or batteries.

3 Reactive Executable Markers

Informed by the related work introduced above, we propose *Reactive Executable Markers* (REMs). We define a REM as a *machine-readable marker that carries multiple executable payloads which are selectively revealed based on the current environment condition.* This behavior can be achieved for instance visually through functional inks (e.g., thermochormic, hydrochromic, etc.) that change their appearance based on the environmental conditions. With REMs, we aim to introduce the idea of "smart markers" that reactively encode information about the condition of the environment into the executable code they expose.

3.1 Superserious: Applications for REMs

Re-using an application proposal for eQR from [8], possible application of REMs may be presented tangibly with an example from the industrial IoT space: A piece of machinery requires troubleshooting. However, the maintenance technician does not have network connection in the machine's location (e.g., around the generator of a hydrothermal power plant) and therefore relies on information available locally. By placing an REM which reacts to temperature and humidity (using thermochromic and hydrochromic inks, respectively), the technician can be effectively guided towards potential sources of the observed failure; for instance, the REM may expose an executable of a cooling system check if the machine temperature exceeds 60°C, or it may expose an executable with a humidity control check if humidity is above 70%.

Another potential application is in the field of search and rescue robotics where the robots do not always have a-priori knowledge of how to react to physical conditions they may encounter in their environment. For example, a REM with ink that is sensitive to volatile organic compounds may instruct the robot not to use internal components that can cause electrical arcing (and hence risk

an explosion). In another scenario, a REM that is coded with thermochromic ink and pasted on a fire protection door may instruct the robot to attempt to open the door only when the door surface is at a temperature below a given threshold. In large-scale automated handling of fruits and vegetables, trays labeled with REMs may instruct type-specific handling for storage and transportation. For example, grapes that are at almost freezing temperature need to be first defrosted before being put in retailing racks. Similar needs exist also in electronics manufacturing—components need to be pre-processed in a different manner before they enter production lines (e.g., depending on temperature, humidity, and pressure the components have been previously exposed to).

All these application scenarios demonstrate how REMs allow the offloading of the *interpretation* of environmental information to the REM designer—i.e., the individual who consciously places the REMs in an environment (e.g., on the fire protection door) or on an object (e.g., on the fruit tray). This frees the REM user (i.e., the human operator, the robot, or the manufacturing system) from itself interpreting the environmental information, which is ecologically valid, as the REM designer should in these cases have better information about the interpretation. In all the above scenarios, the benefit of using REMs is that the conditional instructions are self-contained and hence the control system does not have to rely on network connectivity to fetch the code and information about environmental conditions from a centralized repository. Interoperability between REMs and automation devices can be achieved by adopting a common standard for interactions such as the W3C's Web of Things (WoT) standard [1]. By embedding code that relies on *semantic description of the actions* (see WoT Scripting API [2]) we avoid tight coupling between the designer of the artifacts and the automation systems.

3.2 Supernatural: Integrating Software Code and Physics

Beyond the concrete value that REMs would bring in specific applications, we argue that the REM concept holds fascinating potential for integrating program code and physical context: as of today, programmers may embed dependencies on environmental conditions (e.g., `if (temp > 20) { ... }`) in the program code they create. However, with REMs, these environmentally-dependent parts of the code are not compiled to an executable that links the code to a sensor (in this case, a temperature sensor); rather, they are compiled into the *type of functional ink* (in this case, a thermoresponsive ink) and the *printing pattern* of the QR code. Hence, the QR code, *itself*, becomes the environment sensor because its exposed executable, that is, the remainder of the program, *itself* adapts to the environment condition. In other words, changes in the environment condition lead to changes of the eQR code which in turn leads to *different* code being executed on the reader device.

From another viewpoint, REMs may be viewed as *encodings* of physical quantities. Consider that the "null program" that might be stored on a REM could expose the measured physical quantity itself—this would be a script that simply writes a "20" when the temperature is 20°C and a "23" when it is 23, etc.: `print(temp)`. Viewed from this perspective, REMs that go beyond such null programs expose an encoded version of the underlying physical quantity that put this quantity *in context*. Here, we think of basic encodings such as `if (temp > 27) print("Warm")`—which are merely a thin contextually interpretative layer on top of the plain value—but we consider that this space, and the space of opportunity it opens, is vast (see Section 3.4).

3.3 Demonstrator

Taking our proposal into practice, we implemented a working demonstrator of the REM concept; in this way, we demonstrate that our concept is compatible with QRscript and show a specific application. This demonstrator serves to elicit current limitations and requirements for future work.

Demonstrator Setup. In our demonstrator, we realize the above-hinted simple application `if (temp > 27) print("it's hot!"); else print("hello world!")` through a REM. Note that, instead of printing a value, without any change to our methodology this program could execute other code, as foreseen in QRscript. As per our methodology, this piece of code requires two eQR codes—one displayed for ambient temperatures above 27°C, and one for temperatures below this value. These two eQR codes are created with the software provided in [9]—these are hence fully compatible with the eQR approach. Our low-temperature executable (`print("hello world!")`) is embedded in the eQR code eQR_{low} and the high-temperature executable (`print("it's hot!")`) in eQR_{high}.

Functional Ink. For our REM demonstrator, we used a cheap (ca. 10 USD / 100ml) reversible thermochormic ink[3] that transitions *from black to transparent* at ca. 27°C. Note that this will expose/reveal what is printed underneath. If the paint is exposed to lower temperature, it (gradually) transitions back to black.

Compiling the REM Program. To achieve the desired behavior, i.e., execution of eQR_{low} if temperature is below 27°C and execution of eQR_{high} otherwise, the two eQR codes are printed on paper following this procedure:

(1) Print eQR_{high} using a laser printer and a standard toner for laser printers.
(2) Fully print over (i.e., cover) eQR_{high} with thermochromic ink.[4]
(3) Print eQR_{low} using thermochromic ink next to eQR_{high}.

Given the transition properties of the thermochromic ink we used, this leads to a product (see Fig. 3) where eQR_{low} is clearly visible at temperatures below 27°C and eQR_{high} is revealed when temperatures increase to above 27°C. We refer to the described combination of eQR_{low} and eQR_{high} as our *demonstrator REM*.

Overcoming Challenges. We evaluated our demonstrator REM to see whether the REM concept works as expected. To do this, we took a picture of the REM at room temperature (ca. 22°C) and another picture after the REM was heated (using excess heat from a laptop at ca. 36°C). These two images were passed on to the QRtree

[1] https://www.w3.org/WoT/
[2] https://www.w3.org/TR/wot-scripting-api/

[3] Specifically, https://www.sfxc.co.uk/de/collections/thermochromic-ink/products/thermochromic-screen-printing-ink-black-27-c
[4] In our demonstrators, we "print" the thermochromic ink manually; however, automatic print procedures for thermochromic and other functional inks are well-known.

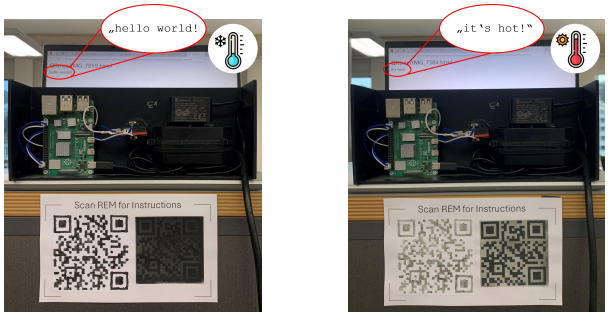


Figure 3: Demonstrator REM under cold (left) and hot (right) condition. Code embedded in eQR_{low} (left eQR code in both pictures) is executed at low temperatures, code embedded in eQR_{high} (right eQR code in both pictures) is executed at high temperatures.

software [9] to execute the embedded eQR codes. This setup did *not* achieve the desired behavior (i.e., execution of eQR_{low} at room temperature and execution of eQR_{high} at higher temperature because the QRtree software failed to identify the eQR codes. This is because the physical properties of our thermocromic ink are imperfect: while the ink is not fully opaque at cold temperatures, it also does not completely vanish when heated. This effect was accentuated by our manual processing of the ink, which resulted in a non-uniform distribution with thicker and thinner layers (see Fig. 3). These challenges could be overcome through the use of higher quality thermochromic ink combined with automatic printing. To realize our demonstrator, we instead compensated for the imperfections by adjusting the QR identification module in the QRtree software to increase its robustness (also under varying lighting conditions). With these adjustments, the execution with our demonstrator REM worked as intended, with the software executing different scripts at low vs. high temperatures (see: Fig. 3 . Thus, we have demonstrated and proven the technical feasibility of the REM concept.

3.4 REM: Limitations and Potential

This paper introduces REMs as machine-readable markers that carry multiple executable payloads which are selectively revealed based on the current environment condition. We have succeeded in demonstrating the proposed concept in practice. However, several limitations remain and point at future design, engineering, and research work. First, our REM demonstrator was hand-painted. This limits the scalability of the concept but could be overcome by using existing automated printing methods for functional inks. Second, our demonstrator does not consider how the transition phase, during which the visual state of the REM is altered, could be handled without causing disruptions to the execution of the payload. Finally, and most importantly, our REM demonstrator—with two eQR codes placed next to one another is inelegant and limited regarding the embeddable programs. While REMs could be designed in a way that makes them look more like single markers, our approach in its basic form is limited to binary `if-else` constructs. This points at research work that would permit the in-place embedding of more than two eQR markers using functional inks, which we hypothesize could be accomplished through careful exploitation of coding properties (specifically redundancy) of the QR standards; if compatibility with QR is not required, it is certainly possible to design a new type of marker from the ground up, with reactivity in mind.

Either of these advances would give rise to the problem of creating a physics-oriented compilation engine that takes program code with environment-dependent instructions (`if (temp > 27) {...}; if (pressure > 1000) {...};` etc.) and compiles these instructions into marker patterns for the appropriate type of environmentally reactive material. This compiler would therefore relate environmental properties expressed in the program code (e.g., using keywords `temp`, `pressure`, etc.) to REM-based physical sensing, realizing a direct integration of program code with physical properties. Next, future work should evaluate demonstrators that realize other types of functional paint beyond thermochromicity and thereby evaluate the transferability of the REM concept to other physical phenomena. To this end, note that REMs as defined in this paper are not bound to remain *visual* markers—since program code may also be transmitted aurally, haptically, or through current modulation, the concept of reactive executables could apply across modalities. Finally, we propose that future work should capitalize on the view of REMs as embedding interpretations or encodings of environmental values. Concretely, we propose that QRscript could be replaced with a binary executable that is suitably encrypted to only be executed in a virtual machine that is provided by the REM designer itself. In this way, entities (e.g., industrial companies) could embed interpretative code in customer environments while retaining this code as trade secret—combining this with our application scenarios from Section 3.1, it would permit run-time contextual interpretation of markers in perishable goods logistics or in electronics manufacturing.

4 Conclusions

In this paper, we proposed REMs as a new type of machine-readable marker that is both reactive to its environment and holds executable program code, thus making the executable itself context-dependent. This introduces the idea of "smart markers" that reactively encode information about the condition of the environment into the executable code they expose; we argue that such encoding can be seen as an *embedded contextual interpretation* of the environmental condition that is tracked by a REM.

We have identified several areas of application which could immediately benefit from the application of REMs (e.g. machine maintenance, reactive robots, logistics, and manufacturing). Beyond these immediate applications, we argue that the REM concept holds fascinating potential for integrating program code and physical context. We furthermore demonstrated a REM prototype and illustrated its usage in a basic mobile robot scenario where a robot's navigation is contextually determined through a REM based on the ambient temperature. Our demonstrators prove the technical feasibility of REMs and uncover several areas in which future work is required or opportune. Furthermore, in general, we hope to inspire research in the field of environment- or object-embedded reactive executables.

References

[1] ISO/IEC 18004. 2015. *Information technology — Automatic identification and data capture techniques — QR Code bar code symbology specification.* Retrieved December 09, 2024 from https://www.iso.org/obp/ui/#iso:std:iso-iec:18004:ed-3:v1:en:en.

[2] GS1. 2024. *50 years of transforming tomorrow.* Retrieved December 09, 2024 from https://www.gs1.org/about/50YearsOfGS1

[3] Dominic Guinard, Joel Vogt, Niall Murphy, Iker Larizgoitia Abad, and Shmuel Silverman. 2021. Dynamic product tag based on an environmental condition. US11068761B2 Patent.

[4] Philipp Gütlich, Yann Garcia, and Theo Woike. 2001. Photoswitchable coordination compounds. *Coordination Chemistry Reviews* 219-221 (2001), 839–879. doi:10.1016/S0010-8545(01)00381-2

[5] Kay Erik Jenss and Simon Mayer. 2023. QRUco: Interactive QR Codes Through Thermoresponsive Embeddings. In *Extended Abstracts of the 2023 CHI Conference on Human Factors in Computing Systems*. 1–4. doi:10.1145/3544549.3583923

[6] Anusha Prabhu, MS Giri Nandagopal, Prakash Peralam Yegneswaran, Hardik Ramesh Singhal, and Naresh Kumar Mani. 2020. Inkjet printing of paraffin on paper allows low-cost point-of-care diagnostics for pathogenic fungi. *Cellulose* 27 (2020), 7691–7701.

[7] Sara Santiago, Miguel Aller, F Javier del Campo, and Gonzalo Guirado. 2019. Screen-printable electrochromic polymer inks and ion gel electrolytes for the design of low-power, flexible electrochromic devices. *Electroanalysis* 31, 9 (2019), 1664–1671. doi:10.1002/elan.201900154

[8] Stefano Scanzio, Gianluca Cena, and Adriano Valenzano. 2022. QRscript: Embedding a Programming Language in QR codes to support Decision and Management. In *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*. 1–8. doi:10.1109/ETFA52439.2022.9921530

[9] Stefano Scanzio, Matteo Rosani, and Mattia Scamuzzi. 2024. *QRtree software*. https://github.com/eQR-code/QRtree

[10] Stefano Scanzio, Matteo Rosani, Mattia Scamuzzi, and Gianluca Cena. 2024. QR Codes: From a Survey of the State of the Art to Executable eQR Codes for the Internet of Things. *IEEE Internet of Things Journal* 11, 13 (2024), 23699–23710. doi:10.1109/JIOT.2024.3385542

[11] Yuan Xu, Zhangming Liu, Rui Liu, Mengxue Luo, Qi Wang, Liqin Cao, and Shuangli Ye. 2021. Inkjet-printed pH-sensitive QR code labels for real-time food freshness monitoring. *Journal of Materials Science* 56 (2021), 18453–18462.