Requirements Cube: Towards a Matrix-based Model of Requirements

Benjamin Aziz[1][a], Gareth Hewlett[2], Ukamaka Oragwu[1][b], Peter Richards[3], Safa Tharib[1][c] and Erica Yang[3]

[1]*School of Creative and Digital Industries, Buckinghamshire New University, United Kingdom*

[2]*Flying River Ltd., United Kingdom*

[3]*Chilton Computing, United Kingdom*

{*benjamin.aziz,ukamaka.oragwu,safa.tharib*}*@bnu.ac.uk,*
gareth@flyingriver.co,
{*peter.richards,erica.yang*}*@chiltoncomputing.co.uk*

Keywords: Healthcare, Matrix Theory, Requirements Engineering, Scenarios, User Stories

Abstract: In critical and robust systems, requirements modeling and analysis is an essential step, called requirements engineering, in the process of system development, which stems from the non-ambiguous identification of end-users and stakeholders needs and goals. Despite the wide application of requirements engineering methodologies, such as KAOS, i*/Tropos, often this step is marred by either the lack of robustness or the lack of usability on part of the analysts. In this paper, we present a 3-dimensional model of requirements, called the Requirements Cube, that is clear, usable and can be manipulated using general matrix algebra. Our model stands on the three main components of requirements; goals, resources and infrastructure, and does not present any complex concepts that may render it unusable. We consider the semantics of the Cube in three different value domains: 2- and 3-valued logic values and probabilistic values. Finally, we demonstrate how this model can be applied to healthcare monitoring scenarios.

1 INTRODUCTION

Despite the widespread use and adoption of requirements engineering methodologies and techniques for the gathering and analysis of stakeholder and end-user requirements, there is still a notable gap in the use of mathematical modeling of such requirements (Yang et al., 2014), except with discrete mathematics-based and logic-based methods (better known as formal modeling). Even so, such formal modeling is not so popular in real-world projects, due to the complexity and the steep learning curve associated with formal methods-based approaches (Bruel et al., 2021). We believe that addressing this integration barrier is essential for advancing the field of requirements engineering and enhancing the quality of software systems and services, in general.

In some critical fields, e.g. healthcare monitoring and care at home for the vulnerable, the application of requirements engineering, gathering and analysis is pivotal for developing effective and user-centered solutions. This process ensures that healthcare technologies are tailored to meet the specific needs of vulnerable (e.g. elderly, disabled) users, thereby enhancing the safety, well-being and overall quality of life for such users (McGee-Lennon, 2008). Moreover, involving end-users in the requirements gathering phase ensures that the developed solutions are user-friendly and address *actual* challenges to those users. For example, recent research (Tajudeen et al., 2022) has demonstrated that understanding requirements from users' perspective is crucial for creating senior-friendly mobile health applications that have increased engagement and lead to better health.

In this paper, we introduce a new mathematical approach, which we term the *Requirements Cube*, for the capturing and modeling of stakeholder requirements[1] and user needs, which is based on matrix theory. We consider that the three most fundamental

[a] https://orcid.org/0000-0001-5089-2025
[b] https://orcid.org/0009-0008-5213-9967
[c] https://orcid.org/0000-0002-0088-3363

[1]We shall use the terms "goal" and "requirement" interchangeably, even though in requirements engineering literature (Van Lamsweerde and Letier, 2002), there is often a distinction between the two, in that a *requirement* is regarded as a refinement of a (high-level) *goal*.

components that underlie the ability to express requirements in any scenario are goals (requirements), resources and infrastructure. Without any of these three components, ones is unable to express fully the requirements of a scenario and how those requirements can be met. Therefore, we introduce a 3-dimensional model that captures these components. We demonstrate how the model can be used to express requirements in a case of a healthcare monitoring scenario inspired from a real-world project, called ADA (ADA Project, 2025). ADA (which stands for "Remote Healthcare Monitoring powered by Network Intelligence and Automation"), focuses on the digital transformation of healthcare using 5G and AI, derisking the adoption of remote healthcare monitoring, with special focus on respiratory condition diagnosis, physiotherapy, and complex care management.

The paper is organised as follows. In Section 2, we discuss some related works. In Section 3, we define the concept of scenarios and their use to extract end-user goals. In Section 4, we present our model, the Requirements Cube, and discuss the concept of a requirements cube, as a three-dimensional matrix that captures end-user goals, required resources and required infrastructure to fulfill those goals. We also consider three cases of how the values of the matrix cells are defined and calculated. In Section 5, we show how a requirements cube can be validated, and whether a specific validation satisfies a predefined fitness function. In Section 6, we discuss the cube's usability. Finally, in Section 7, we conclude the paper and discuss directions for future work.

2 RELATED WORK

Requirements engineering is a crucial phase in the software development process that focuses on identifying, analysing, documenting and maintaining system requirements (Pohl, 2010; Sommerville, 2011). Various approaches to the gathering and analysis of requirements have been developed to address the challenges of capturing stakeholders' needs and ensuring system functionality aligns with business objectives. These include *scenario-based requirements engineering*, *goal-oriented requirements engineering*, *viewpoint-based requirements engineering*, *model-driven requirements engineering* and others.

Scenario-based approaches use real-world scenarios to capture and analyse requirements (Rolland et al., 1998). Such scenarios would provide concrete descriptions of system interactions in order to understand user needs, system behavior and edge cases (Potts et al., 1994). Notable scenario-based methodologies include use case modeling in UML-based software engineering (Cockburn, 2001), as well as event-driven scenario analysis, which aids in defining system responses to external stimuli (Sutcliffe, 2003).

On the other hand, the goal-oriented approach to requirements engineering emphasises more the capturing of stakeholders' objectives and refining these to operational system requirements (Van Lamsweerde, 2001) . The framework provides a structured methodology for decomposing high-level goals into subgoals and constraints, ensuring alignment between stakeholder needs and system capabilities (Dardenne et al., 1993). Popular goal-oriented methodologies include KAOS (Dardenne et al., 1993), $i^\star$ (Yu, 1997) and Tropos (Giorgini et al., 2005).

Viewpoint-based requirements engineering acknowledges the diverse perspectives of stakeholders involved in systems development (Kotonya and Sommerville, 1998). This approach structures requirements based on different *viewpoints*, allowing conflicting interests to be identified and resolved systematically. The VORD (Viewpoint-Oriented Requirements Definition) method (Sommerville et al., 1998) is a well-known viewpoint-based approach that categorises viewpoints into direct users, indirect users, and regulatory authorities. By integrating multiple perspectives, this approach enhances completeness and reduces inconsistencies in requirements.

Formal methods have also been adopted in the area of requirements engineering, giving rise to what is as model-driven requirements engineering (Schmidt, 2006). This approach is closely related to model-driven development in general, where requirements models are transformed into design and implementation artefacts. The use of domain-specific languages and formal verification techniques ensures consistency and correctness in the development lifecycle (France and Rumpe, 2007). Techniques such as SysML (Systems Modeling Language) and UML-based modeling have also been widely applied in this domain (Weilkiens, 2007). Examples of formal language applications in model-driven requirements engineering include Alloy for constraint specification (Jackson, 2012), B-method for rigorous system modeling (Abrial, 1996), and Event-B for formal refinement for action systems (Butler and Yadav, 2012).

In more recent years, the use of AI-driven techniques has been deployed to many of the requirements engineering activities both in industry and innovation. For example, the authors in (Nadeem et al., 2022) used AI techniques to compare requirements gathering and requirement management tools for IoT-Enabled Sustainable Cities. The research compared various techniques for requirements gathering like,

context diagrams, functional decomposition, AS-IS activity models, TO-BE activity models, user stories and mind maps. Their conclusion was that no single tool is universally optimal as each has its strengths and weaknesses depending on the projects needs.

Other approaches include requirements prioritisation like MoSCoW (Must-have, Should-have, Could-have, Won't-have) (Clegg and Barker, 1994) and Quality Function Deployment (QFD), which help stakeholders make informed decisions about requirements importance (Akao, 2024). Finally, the agile requirements engineering, an emerging field, integrates RE into iterative and incremental software development methodologies, ensuring adaptability and responsiveness to changes (Paetsch et al., 2003).

Within the healthcare sector, requirements engineering and analysis has also been suggested as a major phase when developing new systems. For example, the authors in (McGregor et al., 2008) introduce a structured approach to gathering requirements in health information systems using the patient journey modeling approach, or PaJMa models. Using a case study for a local mental health centre, they showed that the PaJMa model improved on requirements detail of traditional methods by including details like non functional requirements and enhanced staff engagement. This approach, increased interest, acceptance of changes and an improved hospital information system. In (Avelino et al., 2014), the authors emphasise the critical role of requirements gathering in customising health information systems for small-scale healthcare facilities. Using a university health service as their case study, they showed that through a detailed understanding of the facility's processes and user needs, the system can be tailored to match the existing manual workflows, ensuring high usability.

Finally, in (Doyle et al., 2011), the authors explore how healthcare information collected through embedded sensors in smart homes can be effectively delivered to the resident older adults. Their study reveals that while residents express strong interest in managing their own health, receiving health-related data results in concerns regarding privacy, data interpretation and usability of technology. Despite these concerns, participants see long-term benefits in self-monitoring, particularly in preventing cognitive decline. The study highlights the need for user-friendly technology, educational support and privacy safeguards to ensure that older adults can confidently engage with health monitoring systems. All of these are candidates to be represented as requirements in any new system set-up.

3 SCENARIOS AND USER STORIES

In this section, we discuss the concepts of a *scenario* and a *user story*, and give an example user story of healthcare at home to motivate our work in the following sections.

A *scenario* is a detailed description of what happens in reality in regards to the problem at hand. In our case, a scenario will describe the health care process, environment and context. For example, this would be the description of the process of caring for an elderly person living at home or in an assisted-living environment or a patient who is currently recovering in a hospital ward, along with the context of that environment in terms of the various resources available to help the person, as well as the actions the carers, nurses, doctors or any other personnel might perform to help the person being considered. Scenarios are usually described using free-style natural languages and will likely contain technical or specialised terminology. Scenarios can be regarded as first-level descriptions of stakeholders' and end-users' needs.

One of the current popular forms of scenarios are *user stories*, which are essentially user-centric scenarios usually of short lengths. User stories are concise, narrative descriptions of how users interact with a system, focusing on their needs and goals. Written from the perspective of the end user, these stories help ensure that development efforts are aligned with user expectations and requirements. They typically follow a simple format:

"As a [type of user], I want [some goal/requirement] for [some reason]."

This approach promotes user-centric design and facilitates iterative development, allowing teams to deliver value incrementally and address real-world problems effectively (Kannan et al., 2019; Turner et al., 2013).

Figure 1 below describes the story of a hypothetical elderly person (user), named Jane, with healthcare requirements described as a story. In this case, we have followed a free-text approach to the story, rather than structured (controlled) text.

From a visual analysis of the story's text, one can derive two sample goals for Jane:

G1: *Jane needs help if she falls*

G2: *Jane's environment temperature must be maintained at* 20°C

For the case of G1, the resource required is a wearable device, which detects movement (or lack of) and also has a big red panic button to summon help. At

	Jane is an old lady who is suffering from arthritis, and rather frail. As an elderly person, Jane wants to receive immediate assistance if she falls down so that she can feel safe and secure at home (including care homes). In addition to that, Jane requires that her environment temperature be maintained at 20°C. In order to fulfill the first goal, a special wearable device is needed that alerts carers or family members when a fall is detected. For the second goal, a heating device, e.g. a continuously functioning boiler or electric heater, is required at her home.

Table 1: A free-text example of a user story.

the same time, the infrastructure needed to support this resource is a radio communication network (e.g. WiFi, 5G, 4G, 3G etc.) and a source of electricity supply to work the radio network and recharge the device's battery. On the other hand, for G2, we find that the resource needed is a central heating system (consisting of a boiler, radiators, thermostat etc.) or some electric heating device (e.g. underfloor heating, electric radiators, heat pumps etc.). At the same time, the infrastructure needed to support this is either a gas/electricity supply[2] or an electricity-only supply.

A scenario or user story produces as a minimum a set of goals, G1, G2, etc. that underlie that a user story (scenario), as shown in Figure 1.

Once the set of goals underlying a user story or scenario has been defined and formulated, we can then define another set, which is the "set of resources" that will provision those goals and enable them to be achieved or maintained. For example, for the goal of maintaining a constant room temperature for the person requiring healthcare, it is necessary to have a temperature-monitoring sensor and some kind of a temperature control unit. These last two are defined

[2]Strictly speaking, a gas-based central heating system also requires an electricity supply, in order to operate the pumps, thermostats etc. We shall refer to this set-up as "gas/electricity" supply, to differentiate it from an electricity-only heating system.

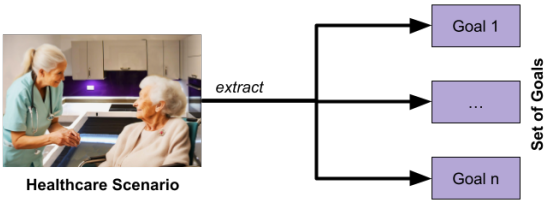

Figure 1: User stories (scenarios) to goals.

as resources that enable our temperature goal. This process is demonstrated in Figure 2.

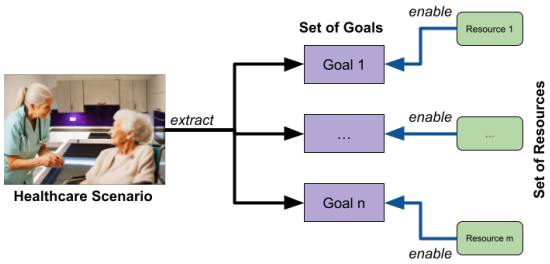

Figure 2: User story goals to resources.

Finally, such resources are deployed on top of "infrastructure", for example, network technologies such as 5G, WiFi etc., as shown in Figure 3.

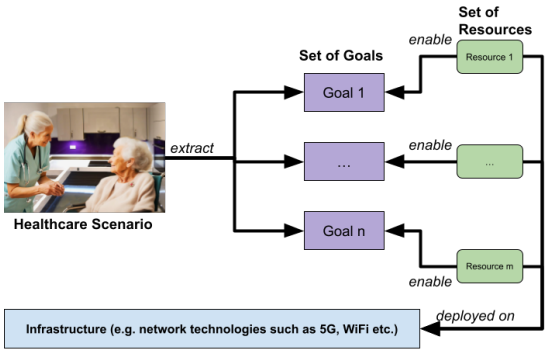

Figure 3: Resources deployment on infrastructure.

It may be worth noting that a resource is within the gift of the user (or whoever is representing their needs) to specify, obtain and deal with. On the other hand, infrastructure elements are provided by 'others' (e.g. energy providers), and therefore are beyond the control of the users.

A typical UML activity diagram for the case of G2 might look something like the diagram of Figure 4.

We next give a formulation of the approach we outlined above in terms of a mathematical model based on matrix theory.

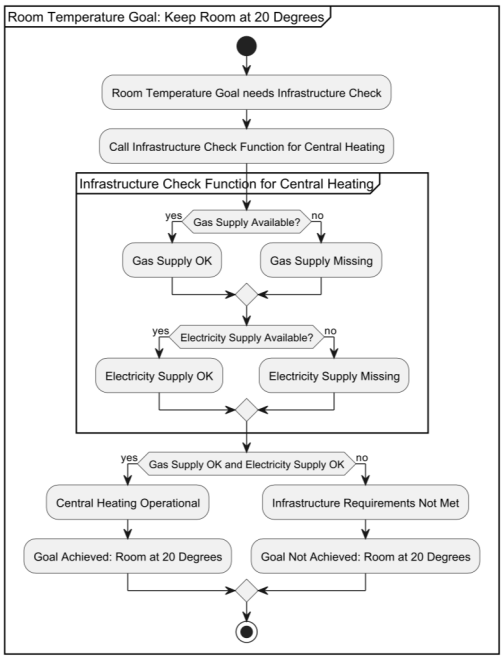

Figure 4: An activity diagram for goal G2 in Jane's story.

4 The Theoretical Model

We start our model by identifying three non-intersecting universal sets:

- $G = \{g_1, g_2, \ldots\}$: the set of all possible goals derived from a scenario, as we discussed in the previous section
- $R = \{r_1, r_2, \ldots\}$: the set of all possible resources, for example any hardware such as various sensors, wearable devices etc., which play a role in the enabling of the goals derived from some scenario
- $I = \{i_1, i_2, \ldots\}$: the set of all possible infrastructure elements, for example energy and communication infrastructure such as gas, electricity, 5G/4G/WiFi etc., which in turn, enable the functioning of the resources required by the goals

We next define the concept of a requirements cube more formally.

4.1 Requirements Cube

A requirements cube is the following mathematical structure, which expresses the relationship between goals, resources and infrastructures.

Definition 1 (Requirements Cube)**.** *Define a requirements cube, A, as a 3-dimensional matrix structure, $A : G \times R \times I$, with elements $(g,r,i) \in A$ such that a resource $r \in R$ and an infrastructure element $i \in I$ enable the fulfillment of a goal $g \in G$.* □

In other words, there is an *association* among elements g, r and i in each cell of a requirements cube. As we explain later in the following sections, the value of the cell will determine the nature of this association. As a matter of analogy, we can imagine the requirements cube above as a loaf of *Battenberg cake* (as in Figure 5), where each *slice* of the loaf represents a 2-dimensional matrix corresponding to the specific goal, g, at which the slice was taken. This analogy brings us to the definition of a requirements slice.

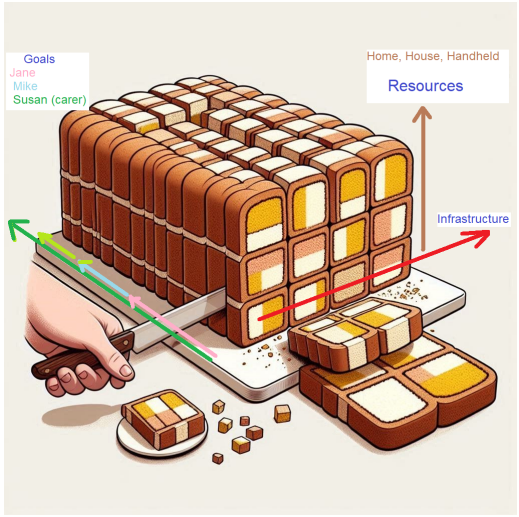

Figure 5: The Battenberg Cake of Requirements.

Definition 2 (Requirements Slice)**.** *For a specific goal, g, define a requirements slice, A_g, as a 2-dimensional matrix, $A_g : R \times I$, with elements $(r,i) \in A_g$ such that a resource $r \in R$ and an infrastructure element $i \in I$ enable the fulfillment of the specific goal $g \in G$, at which the slice is taken.* □

Therefore, a requirements cube is the stacking of several requirements slices together, such that the resulting cube represents the total number of requirements (goals) identified in the context of a specific scenario. In this 2-dimensional matrix (i.e. slice), the values of cells at the intersection of each row and column are calculated based on the specific values of the intersecting rows and columns. We shall next expand on cell values and what that means.

Figure 6 illustrates a generic k-size requirements cube, which stacks together k number of requirements, each requirement slice is defined as an $m \times n$ matrix, with m number of infrastructure elements, and n number of resources. We assume that a cube represents the requirements of a single stakeholder or end-user in whatever scenario is being considered.

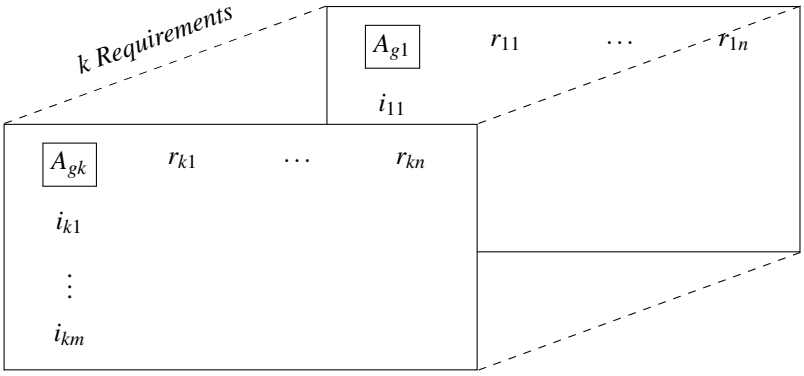

Figure 6: A generic k-size requirements cube.

4.2 2-Valued Logic

In our first case, we assume a model based on 2-valued (i.e. Boolean) logic, the value of a cell in a 2-dimensional slice matrix (i.e. the intersection of a row representing a resource and a column representing some infrastructure) denotes the availability requirement of the resource and infrastructure elements.

We assume that the availability value for a resource or an infrastructure element is defined using the following operation:

$$\nu : (R \cup I) \rightarrow \mathbb{B}$$

Informally, when $\nu(r) = \mathbb{T}$ or $\nu(i) = \mathbb{T}$ for some resource r and infrastructure i, then that means that r and i are *required to be available*, in whatever goal slice they fall within. This is different from the *actual availability* at the environment itself, e.g that there is currently a telephone device available in the patient's apartment (we discuss this difference later in Section 5). On the other hand, if $\nu(r) = \mathbb{F}$ or $\nu(i) = \mathbb{F}$, then this means that either of these two elements is not required to be available, for the current goal to succeed.

The value of a cell in a slice belonging to some goal g, is then calculated as the logical conjunction of the values of $\nu(r)$ and $\nu(i)$. More formally,

$$A_g(r,i) = \nu(r) \wedge \nu(i)$$

Mathematically, this would become a cell in the actual cube by attaching to it the name of a specific goal:

$$A(g,r,i) = (g, \nu(r) \wedge \nu(i))$$

Informally, when $A_g(r,i) = \mathbb{T}$, then this means that the goal for which this slice of the loaf belongs requires both the resource r and the infrastructure element i to be available in order for the goal to be fulfilled. On the other hand, if $A_g(r,i) = \mathbb{F}$, then the goal does not depend on the combination of the specific resource and infrastructure.

Example 1. *for the goal g_2 (from Section 3) that states that "Room temperature must be maintained at $20°C$", and that r = Central Heating, i_1 = Gas/Electricity and i_2 = WiFi, where $\nu(r) = \mathbb{T}$, $\nu(i_1) = \mathbb{T}$ and $\nu(i_2) = \mathbb{F}$, then we have the following matrix:*

$$A_{g_2} = \begin{bmatrix} & \textit{Central Heating} & \ldots & \textit{Falls Alarm} \\ \textit{Gas/ Elect.} & \mathbb{T} & & \mathbb{F} \\ \vdots & & & \\ \textit{WiFi} & \mathbb{F} & & \mathbb{F} \end{bmatrix}$$

Since $\nu(r) \wedge \nu(i_1) = \mathbb{T}$ but $\nu(r) \wedge \nu(i_2) = \mathbb{F}$. Informally, to maintain room temperature at $20°C$, we need the combination of a central heating system (resource) and a gas/electricity (infrastructure) to be both available. But the combination of a central heating system and WiFi (different infrastructure) would not be useful for such goal regardless of their availability. Similarly, the combination of a falls alarm device and either infrastructure element is false since this device is not required for this specific goal, i.e. $\nu(\textit{Falls Alarm}) = \mathbb{F}$.

On the other hand, consider the goal g_1, which states that "the person must be wearing a WiFi-connected falls alarm device", then the matrix for this goal is the following:

$$A_{g_1} = \begin{bmatrix} & \textit{Central Heating} & \ldots & \textit{Falls Alarm} \\ \textit{Gas/ Elect.} & \mathbb{F} & & \mathbb{F} \\ \vdots & & & \\ \textit{WiFi} & \mathbb{F} & & \mathbb{T} \end{bmatrix}$$

where the falls alarm device now is required, in addition to the requirement for a WiFi connectivity el-

ement (e.g. an enabled access point). However, for this goal, neither the central heating resource nor the gas/electricity system are required. The full cube for the above two matrices is as shown in Figure 7. □

4.3 3-Valued Logic

In the second model, we assume a 3-valued logic, where the possible values are $\{\mathbb{T}, \mathbb{F}, \perp\}$, and which include the two Boolean values and an undefined element, $\perp$. The latter expresses situations where the availability requirement on a resource or an infrastructure is simply undefined or unknown. In other words, we do not know if the resource of infrastructure are required for a specific goal. Such 3-valued logic is useful in expressing *incomplete requirements*, where certain values are left undefined in the case of intermediate versions of the requirements cube, but that later become fully defined in the final version.

We adopt the following truth values for the undefined case:

$$\mathbb{T} \wedge \perp = \perp$$
$$\mathbb{F} \wedge \perp = \mathbb{F}$$

For example, if at the stage of specifying the requirements, we are unable to determine the type of the falls alarm's connectivity (i.e. whether it operates over 5G, 4G, WiFi etc.), then it becomes unknown whether WiFi is really required. Hence, $\nu(WiFi) = \perp$, and the 2-dimensional matrix for g_1 becomes as follows:

$$A_{g_2} = \begin{bmatrix} & \textit{Central Heating} & \dots & \textit{Falls Alarm} \\ \textit{Gas/ Elect.} & \mathbb{F} & & \mathbb{F} \\ \vdots & & & \\ \textit{WiFi} & \mathbb{F} & & \perp \end{bmatrix}$$

This means that we are unable to confirm whether a WiFi access point will be required or not at the patient's or vulnerable person's premise, therefore, this requirement remains incomplete at this stage, until further clarification is made on the type of the alarm.

4.4 Probabilistic Availabilities

The third model we consider here is that of probabilistic availability values instead of binary ones. For this we assume a new definition of the ν operation, which we call ν_P:

$$\nu_P : (R \times I) \rightarrow [0,1]$$

Unlike ν, ν_P returns a value between 0 and 1, where 0 denotes the case where the requirement states that the probability of a resource or infrastructure element, x, being available need only be 0% (i.e. that the element is not required to be available. This is similar to saying $\nu(x) = \mathbb{F}$). On the other hand, a value of 1 denotes that the probability is 100%, or in other words, that the element must be available as a requirement (or that $\nu(x) = \mathbb{T}$).

Informally, ν_P represents the *minimum probabilistic expectation* of the availability of some resource or infrastructure at the environment being considered. For example, we may state that the WiFi coverage is expected to be at least 0.8, which means that 80% of time the WiFi signal must be available in some location, or that the WiFi signal may be available in at least 8 out of 10 locations at any point in time (the semantics of this percentage maybe further refined considering various different scenarios). Similarly, it is not uncommon for energy providers to advertise energy supply availability ratios usually the 6 9s golden standard, i.e. 99.9999%, which allows only 31.5 seconds of downtime per year. Again, this would express the availability requirement (as a probability) needed by various devices perhaps stemming from the nature of equipment in those devices or the criticality of their role in the business context.

Based on this probabilistic model, the value of a cell can now be defined as follows:

$$A_g(r,i) = \nu_P(r).\nu_P(i)$$

and this again can be transformed into a cube cell value by including the name of the goal:

$$A(g,r,i) = (g, \nu_P(r).\nu_P(i))$$

Example 2. *Let's consider the same scenario as discussed in Example 1, except this time, we assign probabilistic values to the availability expectation for all the resources and infrastructure elements. For example,* $\nu_P(\textit{Central Heating}) = 0.85$ *and* $\nu_P(\textit{Gas}) = 0.999999$ *for goal* A_{g_2}*, whereas* $\nu_P(\textit{Falls Alarm}) = 0.7$ *and* $\nu_P(\textit{WiFi}) = 0.9$ *for the goal* A_{g_1}*. As a result, we obtain the following two requirement slices:*

$$A_{g_2} = \begin{bmatrix} & \textit{Central Heating} & \dots & \textit{Falls Alarm} \\ \textit{Gas/ Elect.} & 0.84999915 & & 0 \\ \vdots & & & \\ \textit{WiFi} & 0 & & 0 \end{bmatrix}$$

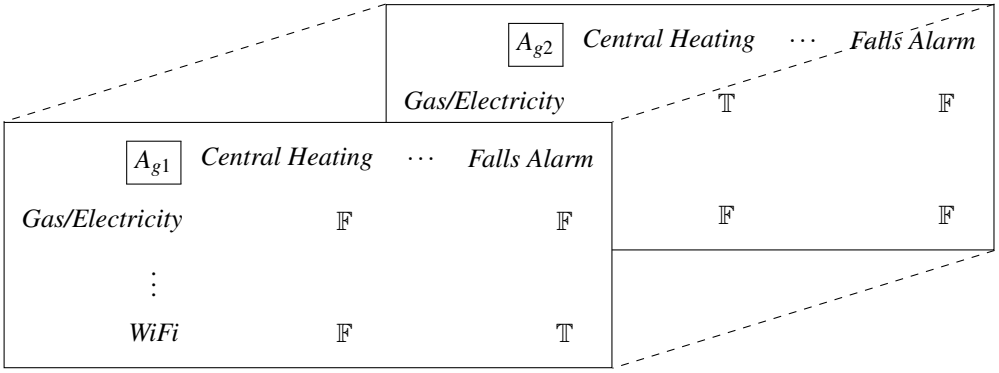

Figure 7: Example of a requirements cube.

$$A_{g_1} = \begin{bmatrix} & \textit{Central Heating} & \dots & \textit{Falls Alarm} \\ \textit{Gas/ Elect.} & 0 & & 0 \\ \vdots & & & \\ \textit{WiFi} & 0 & & 0.63 \end{bmatrix}$$

□

5 REQUIREMENTS CUBE VALIDATION

A requirements cube, specified using some scenario, can be *validated* in terms of the actual real-time availability values for the resources and infrastructure elements in some environment. This validation step will clarify whether the requirement (slice) is being me or not, and therefore, if we need to take further mitigation steps to remedy the situation.

Define the following operation, which returns the *actual* availability value for some resource or infrastructure, at a certain point in time:

$$\eta : (R \cup I) \to \mathbb{B}$$

Note that η is different from ν in that the latter expresses the availability *requirement* whereas the former expresses the *actual* availability value.

Based on the above, we define a *2-valued fitness function* as one that compares the specified availability to the actual availability values, within the 2-valued logic we discussed in Section 4.2:

$$f : \mathbb{B} \times \mathbb{B} \to \mathbb{N}$$

We define f as follows:

$$f(x,y) = \begin{cases} 0 & \textit{if } x = \mathbb{T} \textit{ and } y = \mathbb{F} \\ 1 & \textit{otherwise} \end{cases}$$

where $\forall e \in (R \times I) : x = \nu(e)$ is the expected (required) availability value for some resource or infrastructure element, e. On the other hand, $\forall e \in (R \times I) : y = \eta(e)$ is the actual availability value for that element e. The fitness function will return 0 only in the case where the expected (required) value was true (i.e. requiring an available element), whilst the actual value was false (i.e. element was not available). A 0 value represents an *unfit* requirement, whereas a 1 value represents a *fit* requirement.

To simplify the visual representation of the fitness function, We adopt the following colouring scheme:

A fit requirement

An unfit requirement

For example, in our scenario, if the the WiFi connectivity is found, on a certain day/time, to be completely down at Jane's home, whereas the gas/electricity supply is running as normal and both the falls alarm and the central heating resources are functional, then her validated requirements cube becomes as in Figure 8, due to the fact that the fitness of A_{g_1} is 0 (unfit). We took the liberty to also colour the specific cell, which is causing the fitness of the requirement slice to be 0. In an actual situation, such a visual scheme will indicate to the care provider that a certain requirement is not being met at the patient's premise or environment.

6 THE CUBE'S USABILITY

So far, our discussion has focused on a single user only, Jane. The real expressive power of the requirements cube technique comes when we consider more end-users (Mike, Stacey etc.), and other stakeholders, e.g. carers (Susan, Steve etc.) who are part of the same environment or in an environment connected or related to the current one.

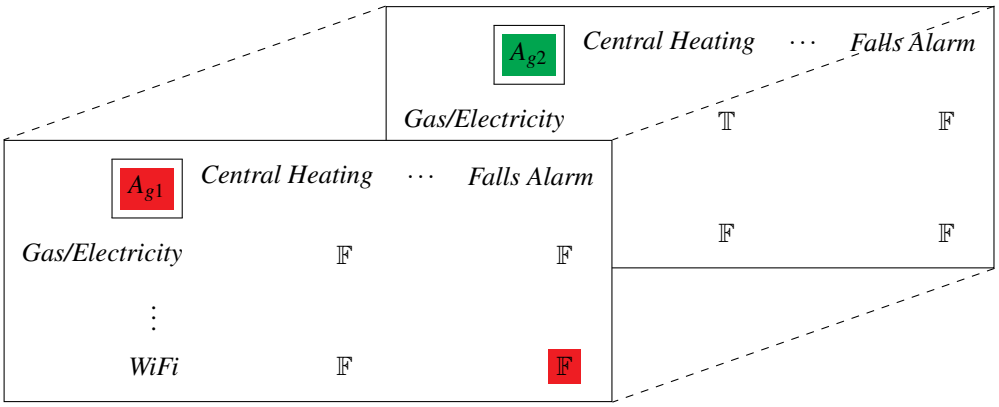

Figure 8: An example of a validated requirements cube.

It is possible to model a single healthcare *ecosystem* (or environment) as one huge cube, which is the composition of the individual users' cubes. If we assume that Jane's requirements cube is called A_{Jane} : $G_{Jane} \times R_{Jane} \times I_{Jane}$ and her carer's requirements cube is A_{Susan} : $G_{Susan} \times R_{Susan} \times I_{Susan}$, then we can model the full set of requirements as a composite cube, as shown in Figure 9, which would also include Mike's cube, where Mike is another vulnerable person being looked after Susan, in the same neighborhood as that of Jane. This may be akin to putting three identical Battenberg cakes on the table, one behind the other. The resulting cake is bigger, but the coloured elements are in the same relation to each other.

We now discuss a few scenarios, which might constitute ground for future research work.

6.1 Failure Scenarios

It is important for evaluating care actions and risks to consider failure scenarios. In Example 2, we assigned a probabilistic availability expectation on the WiFi connectivity to be $\nu_P(WiFi) = 0.9$. However, suppose that the WiFi fails, i.e. $\eta(Wifi) = 0$, at a given instant in time. Then Jane, Susan or anyone else connected, will suddenly have a problem with their WiFi-dependent requirement(s) identified simultaneously by one or more slices of the composite cube matrix becoming unfit (i.e. turning red). Appropriate care actions, therefore, can follow for all persons affected. However, this requires an adequate risk mitigation plan and an underlying risk analysis of the damage that a requirement's slice turning red can cause.

6.2 Scenarios which Change with Time

The cube, including need for the goals, and the availability of resources and infrastructure has been considered at a specific moment in time, and in the context of some static scenario. However, change may be introduced to scenarios, with time. For example, Jane may have a relative staying with her, so while her goal of getting up if she falls remains, the resource requirement of a wearable device is reduced (or probably even eliminated). Likewise, if Mike is in hospital for a few days, his requirements will also change due to increased and different level of care being available at the hospital, compared to his home environment.

As a result, one of the future direction of work stemming from this paper will focus on the adaptation of the requirements cube with time and with the change of scenarios (including changes in requirements, resources and infrastructure).

6.3 Role of Susan the Carer

As we mentioned earlier, the discussion thus far had centred on single end-user clients, e.g. Jane, and then this was expanded to include multiple clients and other stakeholders, by creating composite cubes.

However, it is also possible that the added stakeholders, e.g. Susan the carer, might have a set of requirements that includes her clients requirements, but also her own unique requirements (e.g. respecting client time slots and traveling around to complete visits at different clients' residences). Since Susan has to look after the clients, the goals of the clients will also be goals for Susan. Therefore, adding new stakeholders' cubes may require revisiting the added requirements' slices to remove redundancies and ensure consistencies. In this way the commonalities and dependencies between the requirements can be interlinked, and changes over time readily understood.

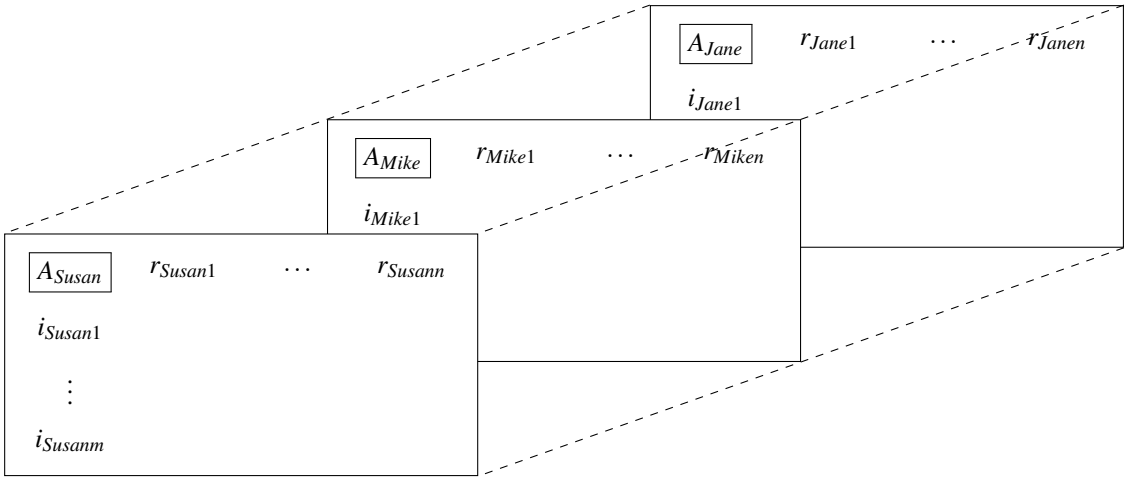

Figure 9: A composite cube, representing the requirements for Jane, Mike and Susan.

7 CONCLUSION

We presented in this paper the sketch of a new model of end-user requirements based on matrix theory. We termed this model the Requirements Cube due to its analogy with Battenberg cake loaves. We defined a slice in this loaf to be a single requirement, with rows and columns representing relevant resources and infrastructure elements. The intersection value of each cell defines the semantics of whether the resource and infrastructure elements are needed for the specific requirement. The stacking of multiple requirements then defines a full loaf (cube).

It would be interesting, in the future, to explore further the current ideas in a digital twin and the creating of training data to support the validation of this new model. In particular, we plan to apply the model to several scenarios observed within the current project ADA (ADA Project, 2025). Additionally, there are several other directions of research work that we highlighted in Section 6, which we plan to pursue in the future. These include the idea of applying machine learning techniques to predict *when* requirements might fail (i.e. become unfit) *before* the conditions for their failure occur. We will use the data collected in project ADA (ADA Project, 2025) as ground truth to train our classification algorithms.

ACKNOWLEDGMENTS

The research work presented in this paper was funded by Innovate-UK grant #10098971 under project name ADA-UK: “Remote Healthcare Monitoring powered by Network Intelligence and Automation”.

REFERENCES

Abrial, J.-R. (1996). *The B-Book: Assigning Programs to Meanings*. Cambridge University Press.

ADA Project (2025). https://www.celticnext.eu/project-ada/. Last accessed 1 February 2025.

Akao, Y. (2024). *Quality function deployment: integrating customer requirements into product design*. CRC Press.

Avelino, J. N. M., Hebron, C. T., Laranang, A. L. E., Paje, P. N. G., Bautista, M. M. F., and Caro, J. D. (2014). Requirements gathering as an essential process in customizing health information systems for small scale health care facilities. In *IISA 2014, The 5th International Conference on Information, Intelligence, Systems and Applications*, pages 184–189. IEEE.

Bruel, J.-M., Ebersold, S., Galinier, F., Mazzara, M., Naumchev, A., and Meyer, B. (2021). The role of formalism in system requirements. *ACM Computing Surveys (CSUR)*, 54(5):1–36.

Butler, M. and Yadav, D. (2012). An overview of formal methods tools and techniques for model-based systems engineering. *Innovations in Systems and Software Engineering*, 8:223–229.

Clegg, D. and Barker, R. (1994). *Case method fast-track: a RAD approach*. Addison-Wesley Longman Publishing Co., Inc.

Cockburn, A. (2001). *Writing Effective Use Cases*. Addison-Wesley.

Dardenne, A., van Lamsweerde, A., and Fickas, S. (1993). Goal-directed requirements acquisition. *Science of Computer Programming*, 20:3–50.

Doyle, J., O’Mullane, B., O’Hanlon, A., and Knapp, R. B. (2011). Requirements gathering for the delivery of healthcare data in aware homes. In *2011 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth) and Workshops*, pages 254–257. IEEE.

France, R. and Rumpe, B. (2007). Model-driven development of complex software: A research roadmap.

In *Future of Software Engineering (FOSE'07)*, pages 37–54. IEEE.

Giorgini, P., Mylopoulos, J., and Sebastiani, R. (2005). Goal-oriented requirements analysis and reasoning in the tropos methodology. *Engineering Applications of Artificial Intelligence*, 18(2):159–171.

Jackson, D. (2012). *Software Abstractions: Logic, Language, and Analysis*. MIT Press, revised edition edition.

Kannan, V., Basit, M. A., Bajaj, P., Carrington, A. R., Donahue, I. B., Flahaven, E. L., Medford, R., Melaku, T., Moran, B. A., Saldana, L. E., et al. (2019). User stories as lightweight requirements for agile clinical decision support development. *Journal of the American Medical Informatics Association*, 26(11):1344–1354.

Kotonya, G. and Sommerville, I. (1998). *Requirements Engineering: Processes and Techniques*. Wiley.

McGee-Lennon, M. R. (2008). Requirements engineering for home care technology. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1439–1442.

McGregor, C., Percival, J., Curry, J., Foster, D., Anstey, E., and Churchill, D. (2008). A structured approach to requirements gathering creation using pajma models. In *2008 30th annual international conference of the IEEE engineering in medicine and biology society*, pages 1506–1509. IEEE.

Nadeem, M. A., Lee, S. U.-J., and Younus, M. U. (2022). A comparison of recent requirements gathering and management tools in requirements engineering for iot-enabled sustainable cities. *Sustainability*, 14(4):2427.

Paetsch, F., Eberlein, A., and Maurer, F. (2003). Requirements engineering and agile software development. In *WET ICE 2003. Proceedings. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.*, pages 308–313. IEEE.

Pohl, K. (2010). *Requirements Engineering: Fundamentals, Principles, and Techniques*. Springer.

Potts, C., Takahashi, K., and Anton, A. I. (1994). Inquiry-based requirements analysis. In *Proceedings of the 16th International Conference on Software Engineering*, pages 58–76.

Rolland, C., Souveyet, C., and Achour, C. (1998). Guiding goal modeling using scenarios. In *Proceedings of the IEEE International Symposium on Requirements Engineering*, pages 53–62.

Schmidt, D. C. (2006). Model-driven engineering. *IEEE Computer*, 39(2):25–31.

Sommerville, I. (2011). *Software Engineering*. Addison-Wesley.

Sommerville, I., Sawyer, P., and Viller, S. (1998). Viewpoints for requirements elicitation: A practical approach. In *Proceedings of the International Conference on Software Engineering (ICSE)*, pages 74–81.

Sutcliffe, A. (2003). Scenario-based requirements engineering. *Proceedings of the IEEE International Requirements Engineering Conference*, pages 320–329.

Tajudeen, F. P., Bahar, N., Tan, M. P., Peer Mustafa, M. B., Saedon, N. I., and Jesudass, J. (2022). Understanding user requirements for a senior-friendly mobile health application. *Geriatrics*, 7(5):110.

Turner, A. M., Reeder, B., and Ramey, J. (2013). Scenarios, personas and user stories: User-centered evidence-based design representations of communicable disease investigations. *Journal of biomedical informatics*, 46(4):575–584.

Van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour. In *Proceedings fifth ieee international symposium on requirements engineering*, pages 249–262. IEEE.

Van Lamsweerde, A. and Letier, E. (2002). From object orientation to goal orientation: A paradigm shift for requirements engineering. In *International Workshop on Radical Innovations of Software and Systems Engineering in the Future*, pages 325–340. Springer.

Weilkiens, T. (2007). *Systems Engineering with SysML/UML: Modeling, Analysis, Design*. Morgan Kaufmann.

Yang, Z., Li, Z., Jin, Z., and Chen, Y. (2014). A systematic literature review of requirements modeling and analysis for self-adaptive systems. In *Requirements Engineering: Foundation for Software Quality: 20th International Working Conference, REFSQ 2014, Essen, Germany, April 7-10, 2014. Proceedings 20*, pages 55–71. Springer.

Yu, E. S. (1997). Towards modelling and reasoning support for early-phase requirements engineering. In *Proceedings of the 3rd IEEE International Symposium on Requirements Engineering (RE'97)*, pages 226–235.