Sanat ve Tasarım Dergisi, 2025; (36): 205-233 / Araştırma Makalesi
e-ISSN 2149 - 6595

USING FLOWCHARTS AND STATE DIAGRAMS AS TOOLSFOR INDUSTRIAL DESIGN PLANNING OF SMART PRODUCTS

AKILLI ÜRÜNLERIN ENDÜSTRIYEL TASARIMININ PLANLANMASINDA AKIŞ VE DURUM DIYAGRAMLARININ ARAÇ OLARAK KULLANIMI

Doç. Dr. Hüseyin Özkal Özsoy
Mimar Sinan Güzel Sanatlar Üniversitesi,
Mimarlık Fakültesi, Endüstriyel Tasarım Bölümü
ozkal.ozsoy@msgsu.edu.tr
ORCID ID: **0000-0001-5531-3539**
MAKALE GELİŞ TARİHİ: **14 Mayıs 2025** · YAYIMA KABUL TARİHİ: **12 Haziran 2025**

Abstract

This study investigates the integration of flowcharts and state diagrams into industrial design, emphasizing their potential for design planning, modeling smart product behavior, and facilitating interdisciplinary communication. While these tools are commonly utilized in engineering and software, their application in industrial design is explored through a case study of a smart desk lamp. The process starts with storyboards for user scenarios, followed by a comparison of flowcharts and state diagrams. The tools are evaluated based on interaction counts, time spent, and structure. Results indicate that flowcharts effectively support sequential planning, while state diagrams more accurately represent condition-based behavior. Their combined use provides both creative flexibility and technical clarity.

Key Words: *Flowcharts, State Diagrams, Industrial Design Process, Smart Product Behavior, Interdisciplinary Collaboration*

Öz

Bu çalışma, akış ve durum diyagramlarının endüstriyel tasarım süreçlerine entegrasyonunu inceleyerek, bu araçların tasarım sürecinin planlanması, akıllı ürün davranışlarının modellenmesi ve disiplinler arası iletişimi desteklemedeki potansiyel katkılarını ortaya koymaktadır. Genellikle mühendislik ve yazılım gibi teknik alanlarda kullanılan bu diyagramların, endüstriyel tasarım bağlamındaki sistematik ve işlevsel kullanımı, çalışmada bir akıllı masa lambasının geliştirilmesine dayalı bir vaka çalışması üzerinden ele alınmıştır. Süreç, hikâye panolarıyla başlayan kullanıcı senaryosu modellemesini takiben, akış ve durum diyagramlarının karşılaştırmalı analizine yönelmiştir. Araçlar; kullanıcı etkileşim sayıları, süre ölçümleri ve diyagram yapısı açısından değerlendirilmiştir. Bulgular, bu iki diyagram türünün birlikte ve tamamlayıcı şekilde kullanımının tasarım sürecine hem yaratıcı düşünce hem de teknik netlik sağladığını göstermektedir.

Anahtar Kelimeler: *Akış Diyagramları, Durum Diyagramları, Endüstriyel Tasarım Süreci, Akıllı Ürün Davranışı, Disiplinlerarası İş Birliği*

• Özsoy, H., Ö. (2025). Using Flowcharts and State Diagrams as Tools for Industrial Design Planning of Smart Products. Sanat ve Tasarım Dergisi, (36), 205-233.

INTRODUCTION

In industrial design, especially when developing interactive or smart products, tools that support structured thinking and clear communication are essential (Self, 2012; Wang et al., 2023). This study investigates the use of flowcharts and state diagrams by comparing them with the more intuitively visual storyboard approach (Simon, 2012), seeking to establish a bridge between visual narrative thinking (Campbell & McDonagh, 2009) and formal system logic (Adelt et al., 2021).

The central research question addressed in this article is: *"How do flowcharts and state diagrams contribute to the industrial design process of smart products compared to storyboards, and in what ways might they be more effective?"* To explore this question, a smart desk lamp was selected as the focus of the case study.

Although diagrammatic tools are widely studied in engineering and software development contexts, their systematic application within industrial design remains relatively underexplored in the literature. This study aims to fill the gap by exploring how these tools can not only support design planning but also serve as a shared visual language to enhance interdisciplinary collaboration (Kim & Lee, 2016; Micheli et al., 2012).

PRODUCT DESIGN PLANNING

Tools that structure thinking and enhance communication play a central role in industrial design, especially when planning smart products with dynamic, user-driven behaviors (Self, 2012). While storyboards have long been used as a visual method for mapping user interactions (Simon, 2012), the increasing complexity of today's smart products has challenged their expressive capacity. This section critically examines three tools—storyboards, flowcharts, and state diagrams—not only in terms of their representational features but also as cognitive scaffolds and shared design languages. Their comparative potential is discussed through the lens of interaction design theory and system modeling, including Norman's execution-evaluation model and user-centered design principles (Norman, 1988).

Storyboards

Originating in film and animation, storyboards are used to depict sequences of interaction through visual narratives (Goldman et al., 2006; Simon, 2012). In design, each frame illustrates a specific moment in a product-use scenario, such as part of a daily routine, often accompanied by annotations that convey user intent or emotion (Truong et al., 2006).

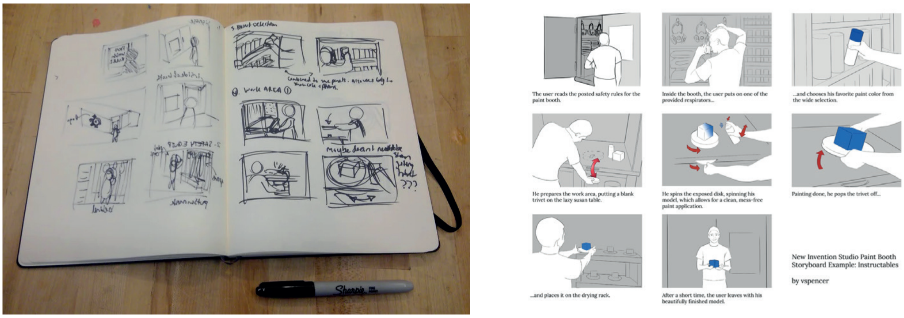


Figure 1: *Storyboarding for Product Design (Spencer, 2025)*

As shown in Figure 1 (Spencer, 2025), storyboards have become common in early-stage product development, helping designers empathize with users, identify context-specific challenges, and frame the problem space (Östlund, 2022; Rodda et al., 2022).

Despite their narrative richness, storyboards lack formal logic, making them less suitable for modeling system-driven behavior. Their static and often linear structure does not accommodate feedback loops, branching scenarios, or algorithmic detail, which are central to smart product functionality (Wang et al., 2024). According to Norman's action cycle, storyboards visualize only the "execution" of user goals, with limited ability to represent the system's internal "evaluation" processes (Norman, 1986, 1988).

Flowcharts

Flowcharts provide a structured method for representing the procedural logic of a system (Weng, 2024). Initially developed for engineering and business applications (Chinofunga et al., 2025; Hossain et al., 2024), they use standardized symbols to depict processes, decisions, and transitions in a clear, step-by-step sequence (Newman, 1998), as shown in Figure 2 (Creately, 2025).

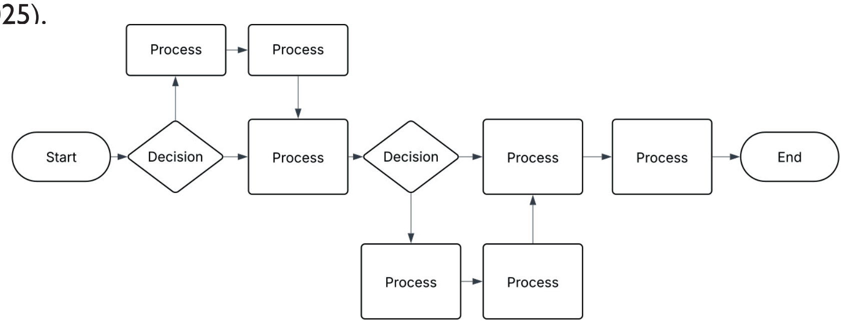
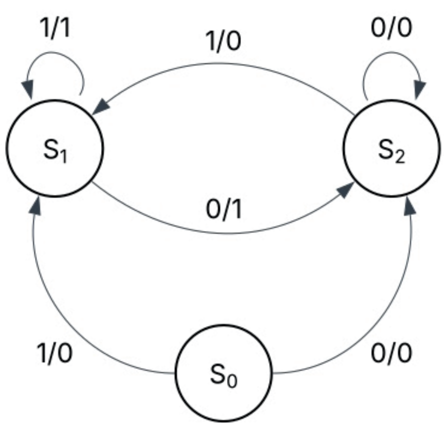


Figure 2: *A sample flowchart that contains various processes and decisions (Creately, 2025)*

From a design cognition perspective, flowcharts serve as externalized mental models, aligning with Norman's concept of "knowledge in the world" (Doyle, 2002). They make procedural logic explicit and testable, facilitating iteration and discussion across team members. While they answer the question, "*What happens next?*", they do not inherently capture the system's current state or its readiness for transition.

State Diagrams

A state diagram models the different states a system operates during its life cycle, also showing the transitions between states and the events triggering those transitions (Olivé, 2007). It begins with an initial state and progresses through various transitions based on defined actions or events (Jacob, 1985), like the possibility of a door being in three distinct states: open, closed, or ajar (Ma & Yu, 1998). Actions such as pushing or pulling the door initiate transitions between these states (Jacob, 1985). These diagrams are rooted in algorithmic thinking, where problems are framed through a logical series of steps, inputs, and outputs (Herman & Choi, 2017). This methodical structure enables the modeling of complex functions (Kant, 1985).




```
state S0 {
    on 1 / output 0 -> S1
    on 0 / output 0 -> S2
}

state S1 {
    on 1 / output 1 -> S1
    on 0 / output 1 -> S0
}

state S2 {
    on 0 / output 0 -> S2
    on 1 / output 0 -> S1
}
```

Figure 3: *Left: State diagram with S0, S1, and S2 states. Each edge is labeled with "j/k", where j is the input and k is the output ("State Diagram," 2025)*

Right: *The UML language representation of the same diagram (by the Author)*

Figure 3 presents a typical example. On the left is a state diagram with S0, S1, and S2 states, where each edge is labeled with "j/k," denoting input and output. On the right is the UML representation of the same model ("State Diagram," 2025). States are illustrated as circles or rounded rectangles, with transitions represented as arrows labeled by the triggering input (Jacob, 1985). Outputs may also be indicated on these transitions (Koskimies & Mäkinen, 1994).

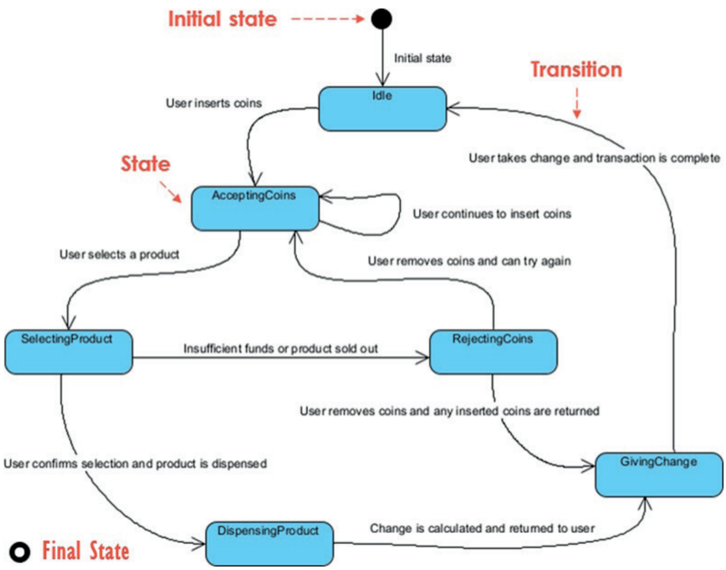


Figure 4: *The state diagram of a vending machine (Vpadmin, 2023)*

State diagrams can be created by hand or with digital tools such as Lucidchart (Diagramming Powered By Intelligence, 2025), Draw.io (Draw.Io, 2025), or UML software (Alhir, 2003), which provide predefined shapes and connectors. Originally used in software and systems engineering, state diagrams remain central in UML for structuring object behavior (Alhumaidan, 2012; Jäger et al., 1999). They are also applied in electrical engineering, robotics, service design, and healthcare, modeling system responses and task transitions as shown in Figure 4 (Bagnati & Del Bello, 2024; Bowman, 2024; Guiochet, 2015; Konrad & Cheng, 2002; Palovuori, 2017; Willett et al., 2018). Generally, a state diagram helps answer, *"What condition is the system in, and what causes it to change?"* (Welte, 2009).

Operation of Flowcharts and State Diagrams

A virtual machine (Goldberg, 1973) running a flowchart executes the diagram steps sequentially, and the program counter continuously increases, pointing to the command being executed as long as the program execution continues (Pavel, 1978). Therefore, flowcharts are generally accepted as more appropriate for planning and documenting design procedures, production steps, or administrative workflows that are inherently sequential (Granfelt, 2017). On the other hand, in a virtual machine running a state diagram, the program counter, which changes millions of times a second, does not necessarily affect the machine state (Millhouse, 2018). The machine waits in a state for a specific input or event to immediately switch to any other state, not necessarily following any order or sequence (Shaw, 1992). This reduces delays in system operations, making state diagrams beneficial in designing smart, connected, interactive, and quick-responding entities (Basheri, 2010).

The planning of design procedures and the specification of product behavior in smart products constitutes a particularly suitable context for the integrated application of flowcharts and state diagrams (Helin et al., 2007). Exploring potential alternative uses for them requires a clear understanding of their differences, allowing industrial designers to choose the most appropriate one based on the specific nature of the tasks they intend to perform (Self, 2012).

Application to Design Process

Industrial design involves structuring design problems to make them more understandable and solvable (Jonassen, 2000). The design process is inherently complex, systematic, and iterative, encompassing multiple phases (Wynn & Eckert, 2017). These phases, outlined in Table 1, include tasks such as problem identification, data collection, concept development, feedback integration, and implementation (Giuliani et al., 2024).

Design Phase	Description
Research	Collecting user needs, trends, and technical constraints.
Ideation	Brainstorming, sketching, and generating alternatives.
Concept Development	Selecting ideas, making mockups, or low-fi prototypes.
Detail Design	CAD modeling, selecting materials, form/function refinement.
Prototyping	Building physical or digital prototypes.
Testing	Usability testing, refinement, and feedback loops.
Finalization	Preparing for production or presentation.

Table 1: *Basic phases of the industrial design process*

The designer can treat each phase of the design process as a “step” or “state”, while transitions represent actions, decisions, or conditions that move him/her from one to another.

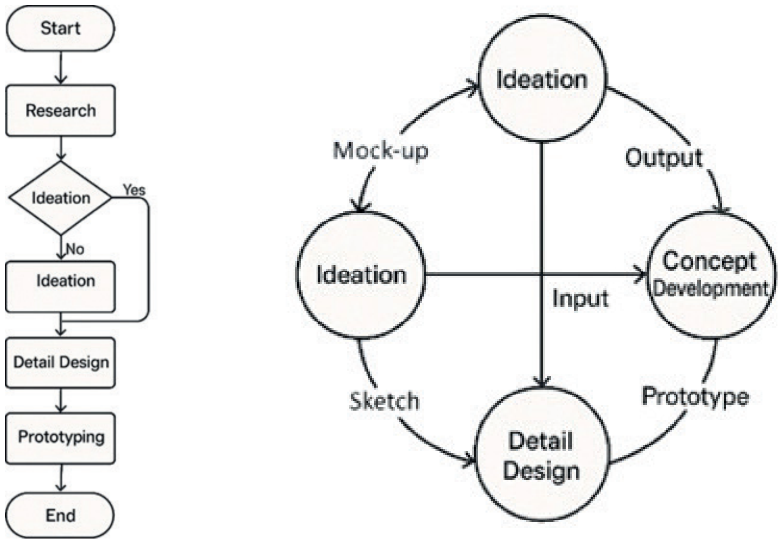


Figure 5: *Flowchart(left) and state diagram(right) of the basic design process (Author's drawings)*

This systematic approach is suitable for utilizing new tools, such as flowcharts and state diagrams, to visualize and test the behavioral aspects of software-based smart products, revealing hidden inconsistencies and assisting teams in making clearer decisions more quickly. The alignment of these tools with software and physical modeling techniques also enhances communication among industrial design, engineering, and programming teams who need to work collaboratively, facilitating more informed cross-disciplinary decision-making during smart product development.

Application to Products Being Designed

If directly applied to product designs, flowcharts, and state diagrams may serve distinct but complementary roles in shaping logic and functionality (Harel, 1987). Flowcharts can be useful for mapping out the internal processes of a product, such as the steps a smart appliance follows when executing a cleaning cycle or the startup routine of an electronic device (Shina, 2012). They can also help identify input/output relationships, operational branches, and sequences that must be logically ordered for the product to function (Vegte & Breemen, 2009).

State diagrams, on the other hand, can model a product's behavior by transitioning between different operational modes in response to contextual changes or user interactions, such as handling interruptions and managing sensory data (Garzon & Louis, 2020). They can be essential for designing functions where transitions do not occur with a fixed sequence but on unscheduled conditions or events, such as waking up from sleep mode, sensing events, error handling, or multi-user settings.

METHODOLOGY

This study adopts an exploratory, qualitative case study approach (R. Ponelis, 2015) to investigate how flowcharts and state diagrams can be used as modeling tools in the industrial design process of smart products. Rather than aiming for statistical generalization (Wah et al., 1995), the research focuses on understanding the functional and cognitive contributions of these tools through a focused design experiment. The experiment was conducted by the author, acting as a single designer, to test how effectively these diagrammatic tools could support process planning, behavior modeling, and interface logic within a compact design cycle.

The design scenario centers on the conceptual development of a smart desk lamp, chosen for its relevance as an everyday product with multiple user interaction points and context-dependent behaviors. This product offers sufficient complexity to model decision logic, sensor input response, and multi-mode operation, making it well-suited to evaluate diagrammatic tools. The methodology involves the creation of storyboards, flowcharts, and state diagrams for three distinct but connected purposes:

- Planning the design process itself.
- Modeling the lamp's functional behavior.
- Designing the user interface logic via a small display on the product.

To evaluate and compare the tools, several empirical metrics were recorded:

- Interaction-based metrics: Number of shape additions, drag-and-drop events, text edits, undo/redo actions, zoom/pan movements, and tool switches.
- Time metrics: Total time spent per tool, average time per iteration, and ratio of idle vs. active time.
- Artifact metrics: Number of states or blocks, labels or annotations, revision passes, and conditional branches or loops.

These metrics were collected manually and via activity logs generated by diagramming software (Lucidchart, 2025; Draw.Io, 2025).

CASE STUDY: DESIGNING A SMART TABLE LAMP

To investigate the practical application of flowcharts and state diagrams, we employed them in the industrial design process of a conceptual smart table lamp. During the study, we analyzed the role of these tools in two key areas mentioned in the previous section: structuring the design process and modeling product functionality.

Storyboarding the Design

We began by storyboarding the design process and functionality of the smart table lamp as two separate sequences of images and notes. This is done on a drawing template, containing a 4-column, 3-row matrix of empty rectangle frames, serving as a visual narrative tool to depict tasks or events. The frames are filled with drawings and notes, each representing a specific moment in time, showing the task in the design process or how the lamp responds to user actions. The two storyboards helped us to visualize the design process and intuitive human-product interaction holistically, clarifying how context and user behavior influence the lamp's operation. We did the drawings in a computer environment, supported by an artificial intelligence (AI) tool, as shown in Figure 6. We provided short text prompts describing the action in each frame. These prompts were input into the AI image generation tool (Vizcom AI, 2025). The AI tool created illustrations that aligned with the described interaction, allowing for rapid visualization without traditional sketching. While this approach speeds up storyboarding by easily creating scenes, the AI input prompts can also be helpful in later design phases to determine product features.

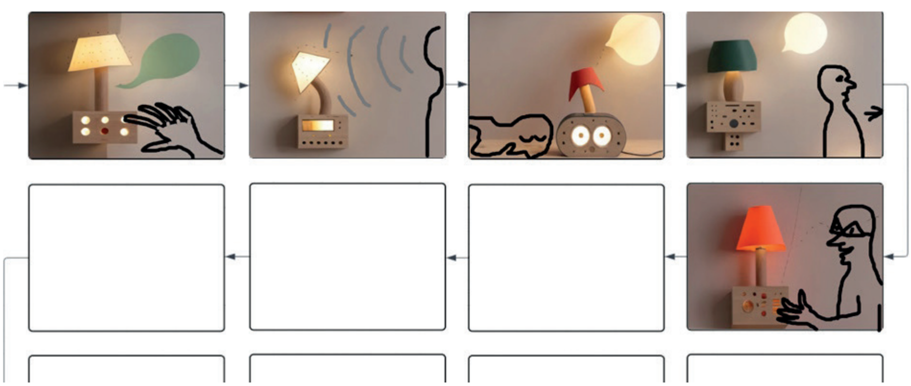


Figure 6: *Sample storyboard representing product functionality, made in Lucidchart, containing images created in vizcom.ai (The actual storyboard images have been omitted from this article.)*

Flowchart of Design Process for the Smart Table Lamp

Table 2 below lists the design tasks, action descriptions, and decision points in the general design process used in this study.

No.	Design Task	Action Description	Conditional Branch / Feedback Loop
1	Define Design Brief	Establish product goals, constraints, and target users	—
2	Conduct User and Context Research	Research usage environments and user preferences	Is user data sufficient? → Yes: Proceed, No: Continue research
3	Identify Functional Requirements	List expected product features (e.g., sensors, manual override, profiles)	Do features align with user needs? → Yes: Proceed, No: Re-prioritize features
4	Map User Scenarios	Write use cases (e.g., evening work, bedtime reading, away from the table)	—
5	Sketch Storyboards	Create visual scenes of user-product interactions	Are interactions clear and intuitive? → Yes: Proceed, No: Revise storyboard
6	Create Initial Flowchart of Behavior	Diagram product logic (e.g., detect motion → light on → adjust brightness)	Are all logical paths and exceptions covered? → Yes: Proceed, No: Add missing logic
7	Develop a State Diagram	Define states and transitions (e.g., Idle, On, Night Mode, Override)	Are all transitions consistent and non-contradictory? → Yes: Proceed, No: Revise
8	Build a Low-Fidelity Prototype	Construct a basic model to test the form and user flow	Does the prototype match the intended logic? → Yes: Test, No: Adjust design or flowchart
9	Conduct Self-Test / Scenario Simulation	Simulate different user cases to identify logic gaps and friction points	Are usability issues found? → Yes: Revise diagrams/prototype, No: Proceed
10	Refine Design	Finalize visual design, logic, and user feedback features	—
11	Document & Prepare for Handoff	Compile diagrams, storyboards, and specifications for development	—

Table 2: *Phases of the smart table lamp design process*

This sequence of tasks is represented as a flowchart drawn in Lucidchart as shown in Figure 7.

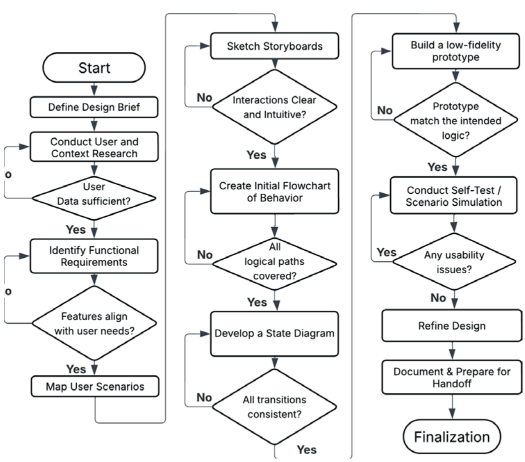


Figure 7: *Flowchart of the smart table lamp design process*

The State Diagram of the Design Process for the Smart Table Lamp

Table 3 lists the design phases as distinct states with transitions triggered by key decisions or milestones, transforming the process into a state machine.

Current State	Event / Trigger	Next State	Notes / Actions
Define Design Brief	Design scope is clearly outlined	User & Context Research	Begin investigating user needs and usage environments
User & Context Research	Sufficient user/context data collected	Identify Functional Requirements	Move from research to translating findings into features
User & Context Research	Data insufficient	User & Context Research	Loop back to continue gathering more data
Identify Functional Requirements	Requirements align with user needs	Map Scenarios & Storyboards	Ready to model usage interactions and context
Identify Functional Requirements	Misalignment found	Identify Functional Requirements	Refine the feature list based on feedback
Map Scenarios & Storyboards	Storyboards illustrate clear interaction logic	Create Behavior Flowchart	Proceed to structure the behavior logically
Map Scenarios & Storyboards	Interactions are unclear or contradictory	Map Scenarios & Storyboards	Revise scenarios for clarity
Create Behavior Flowchart	All logic paths and exceptions are defined	Build the State Diagram	Proceed to model internal system behavior
Create Behavior Flowchart	Gaps in flow or logic	Create Behavior Flowchart	Add missing branches or revise the conditional logic
Build Behavior State Diagram	Valid and complete state transitions	Prototype	Behavior modeling complete—start prototyping
Build Behavior State Diagram	Inconsistent or unclear transitions	Build the state diagram	Revise the state diagram for logical completeness
Prototype	Prototype reflects intended logic	Test & simulate	Ready for testing key user scenarios
Prototype	Gaps between the prototype, charts and diagrams	Refine Flowchart or State Diagram	Adjust the design documentation or prototype behavior
Test & simulate	Usability issues resolved	Refine & Finalize Design	Final refinements ready
Test & simulate	Issues found during testing	Prototype / Chart / Diagram	Loop back as needed, depending on the source of the issues
Refine & Finalize Design	Design is validated	Document & Handoff	Prepare for implementation or presentation
Document & Handoff	Documentation complete	Final State	Process completed, ready for handoff

Table 3: *Smart table lamp design process as a state machine.*

This state machine is represented by the state diagram drawn in Lucidchart, shown in Figure 8.

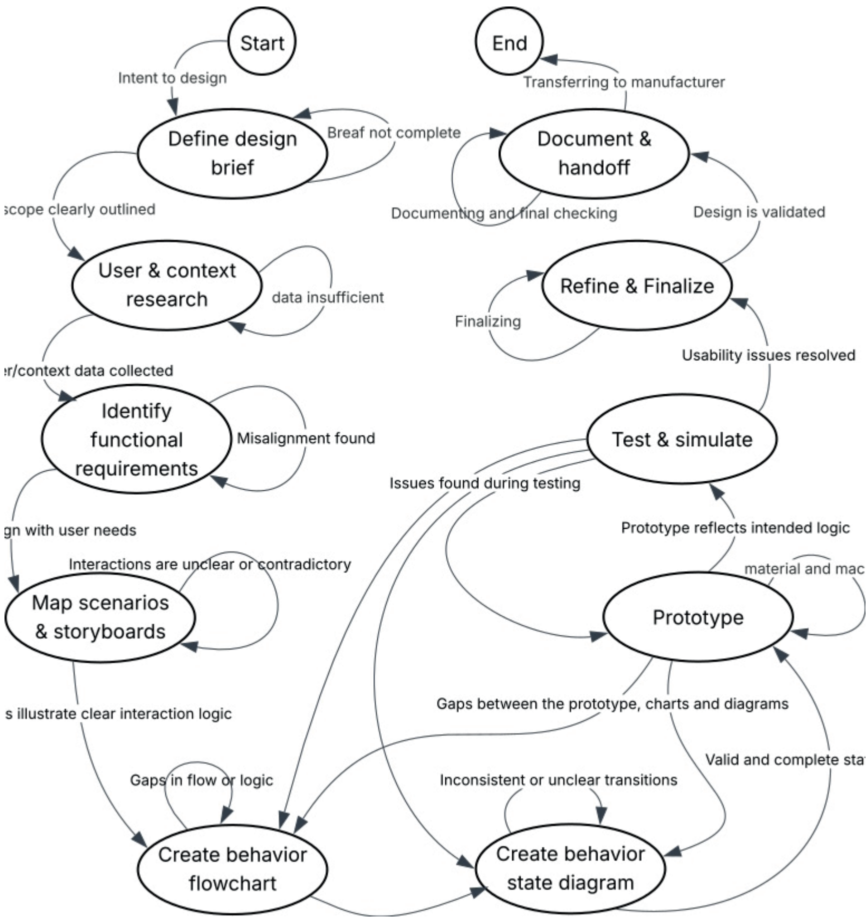


Figure 8: *The state diagram of the smart table lamp design process.*

Design Brief for the Smart Table Lamp

This conceptual product will possess several advanced features that elevate it beyond a mere lighting device, transforming it into an interactive, context-aware system. According to functional requirements determined with the support of user and context research, the lamp:

- Detects ambient light levels and adjusts brightness.
- It has a real-time clock.
- Switches between modes such as day, night, etc., based on time of day or a schedule set by the user.
- Turns on when a user is nearby, turns off or fades out when the user leaves.
- Allows the user to override automation.

• Stores user preferences and puts frequently used functions in quick menu selections.

• May have Internet and Bluetooth connections, accepting input from multiple sources such as mobile apps or voice assistants.

• Can learn user habits over time and adjust automatically.

• Can change this behavioral schedule based on user habits.

• Can download and install new features.

Table 4 below outlines the key functional properties of the smart lamp and evaluates the appropriateness of using flowcharts and state diagrams for modeling each function. The table highlights how both tools offer complementary strengths, and selecting the appropriate one depends on whether the emphasis is on linear logic or responsive system behavior.

Lamp Property	**Purpose/Function**	**Flowchart Use**	**State Diagram Use**
Ambient Light Sensing	Adjusts brightness based on the environment	Defines decision logic for brightness adjustment	Model transitions: *Idle → Adjusting → Stable*
Time-Based Modes (Day/Night/Focus)	Switches modes automatically or on schedule	Outlines scheduled tasks and user-defined rules	Maps states like *Day Mode → Focus Mode → Night Mode*
Presence Detection	Turns on/off based on motion or proximity	Shows conditional logic: *If no motion for X mins → turn off*	Describes state flow: *Off → Sensing → On → Standby*
Manual Touch Override	User interrupts automation to control the light manually	Shows input handling and override procedures	Adds override layer: *Auto Mode → Manual Mode → Auto Resume*
User Profiles / Scenes	Saves and applies the user preferences	Explains steps to create/edit/select a profile	Illustrates behavior shift: *Profile A → Profile B*
App / Voice Assistant Control	External device integration	Represents the flow of commands and system response	Ensures consistent behavior: *Listening → Executing Command → Idle*
Energy Saving Mode	Reduces power after inactivity	Plans inactivity timer and wake conditions	Models sleep logic: *Active → Low Power → Active (on motion)*
Adaptive Behavior	Learns and adjusts based on user habits	Depicts data input and decision points for learning logic	Reflects adaptive state changes over time

Table 4: *Functions of the smart lamp and their suitability for flowchart - state diagram use*

Structuring the Functional Scenario of the Table Lamp as a Flowchart

Table 5 presents a structured sequence of functional steps that define the operational scenario of the smart lamp, modeled in a flowchart format. The scenario begins with system initialization and quickly distinguishes between manual and automatic modes based on user interaction. Conditional logics such as ambient light thresholds, motion detection, and time-of-day evaluations determine whether the lamp turns on, adjusts brightness, or enters energy-saving states.

Step No.	Action / Process	Condition / Decision	Next Step (Yes)	Next Step (No)
1	Start / Initialize System	—	Proceed to Step 2	—
2	Check for Manual Override Input	Was manual mode input received?	Step 3: Activate Manual Mode	Step 4: Check Ambient Light
3	Activate Manual Mode	Wait for user interaction And execute the order	Step 12: Monitor for Exit	—
4	Read Ambient Light Sensor	Is ambient light below the threshold?	Step 5: Turn On the Lamp	Step 6: Enter Idle Mode
5	Turn On the Lamp Automatically	—	Step 7: Monitor Motion	—
6	Enter automatic Mode	Wait for ambient light drop or user input	Step 4: Sensor readings	—
7	Monitor Motion Sensor	Is motion detected?	Step 8: Adjust Brightness	Step 9: Start Inactivity Timer
8	Adjust Brightness Based on Ambient Light	—	Step 10: Evaluate Time of Day	—
9	Start Inactivity Timer	Has time expired (e.g., 10 mins)?	Step 11: Turn Off the Lamp	Loop back to Step 7
10	Evaluate Time of Day	Is the current time within the Midnight Mode schedule?	Activate Night Mode (Dim Light)	Maintain current brightness
11	Turn Off the Lamp	—	Step 4: Wait for the next input	—
12	Monitor for Exit from Manual Mode	Did the user double-tap to exit or timeout?	Step 4: Return to Automatic Mode	Loop in Manual Mode
13	Security mode control	Is there any security breach	Give alarm warnings	Step 4: Return to Automatic Mode
14	Sleep mode control	Is the sleep mode on?	Start sleep timer	Music-play setting control.
15	Music-play setting	Has the sleep timer expired?	Stop music	—

Table 5: *Functional scenario steps of the smart table lamp*

The flowchart drawn in Lucidchart to map these steps as a flowchart is presented in Figure 9.

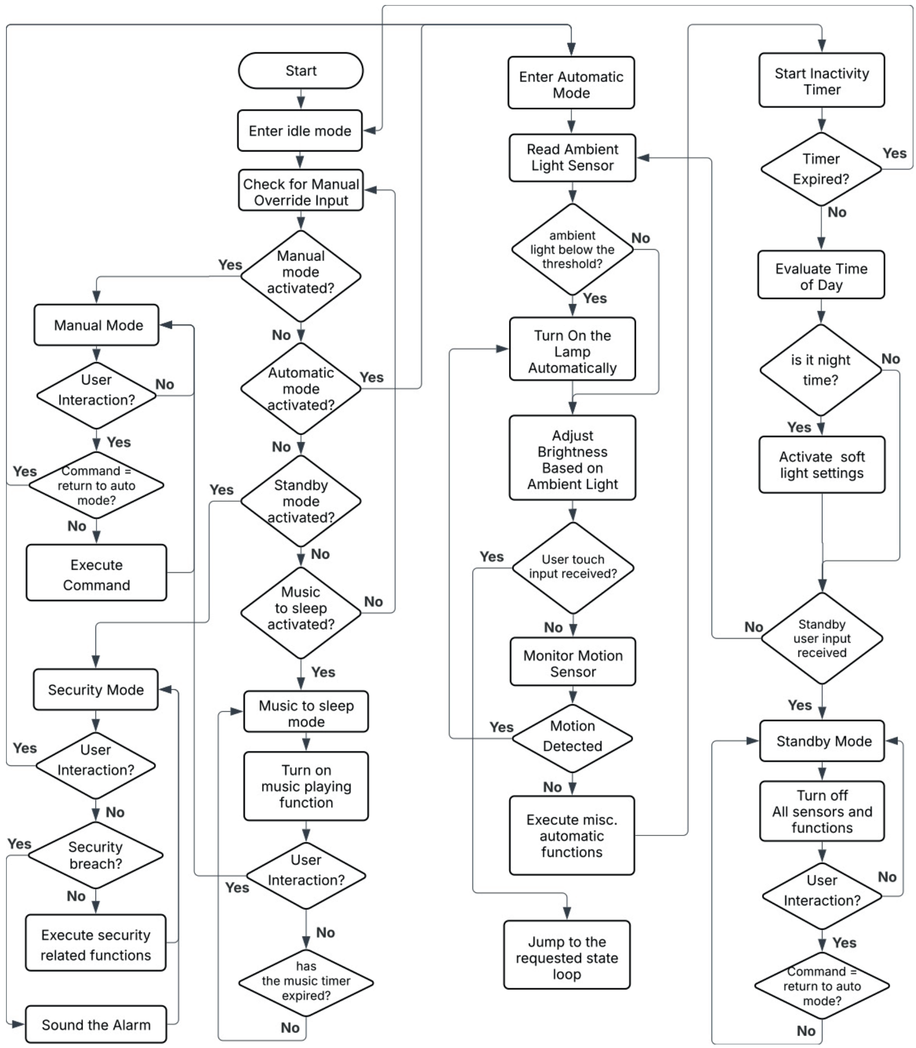


Figure 9: *Functional scenario flowchart of the smart table lamp*

Structuring the Functional Scenario of the Table Lamp as a State Diagram

Operational modes or states of the state diagram are determined as follows.

- *Idle:* A low power, listening state; The sensors and touch display are active, time and date are displayed on the screen, all other functions are off.
- *Manual:* This is a user-controlled state; The lamp functions are adjusted by the user and stay unchanged until a new user input comes, regardless of automated triggers and sensor data.
- *Automatic:* A range of automated tasks are performed

• *Brightness Adjustment according to sensor data*
• *Auto on-off according to user presence detection*
• Fade-in, out, smooth changes, etc., various lighting effects are possible
• *Remote command detection is active*

• *Security:* The lamp doesn't go into standby mode, stays ready for normal use, lighting and screen messages are off, and security sensors are on.

• *Alarm:* A wakeup or security alarm is sounded.

• *To sleep:* The light is on for a time pre-set by the user.

• *Sleep:* The user is asleep. A wake-up alarm can be preset.

• *Stand-by:* A low-power state, entered with a power-down command; all functions, screen, and sensors are off, waiting for the power-up command.

Table 6 defines the operational states of the smart table lamp, the events or inputs that trigger transitions between these states, and the system's corresponding outputs or actions.

Current State	Input / Trigger (Event)	Next State	Output / Action
(Power Off)	Power supply connected / Start	Idle	Initialize sensors and timers
Idle	Auto Touch input detected	Automatic	Switch to automatic operation
Idle	Manual Touch input detected	Manual	Switch to manual control
Idle	No activity for a set time	Stand-by	Nobody home, switch to standby to save power
Automatic	Motion detected	Adjust Brightness	Adapt brightness based on presence and environment
Automatic	No motion for timeout duration	Standby	Dim or turn off the lamp
Automatic	Touch input detected	Manual	Override automation, Enter manual setting
Automatic	If a specific period is preset by the user	Automatic	Set warm, dim lighting
Adjust Brightness	Ambient light > threshold	Automatic	Turn off or dim the lamp
Security	Security breach detected	Alarm	Return to max brightness, perform alarm functions
Alarm	Touch input detected	Manual	The user manually turns off the alarm
Music to sleep	The preset sleep time has passed	Sleep	Fade the light out and stop music/radio
Manual	Touch input (double-tap or exit button)	Automatic	Resume automatic control
Manual	Touch input (via on-screen button)	Idle	Switch to idle mode
Manual	Touch input (via on-screen button)	Security	Switch to security mode
Manual	Touch input (via on-screen button)	Standby	Switch to standby mode
Manual	The user adjusts brightness manually	Manual	Set brightness as per user input
Manual	Inactivity timeout	Sleep	Turn off the lamp

Table 6: *Operational modes or states of the smart table lamp,inputs that trigger state changes, and the corresponding outputs or actions.*

Each state change is linked to a specific input, such as a touch command, a sensor trigger, or a timer expiry, and results in clearly defined actions. This model captures not just logic but also behavioral responsiveness, making it well-suited for systems with contextual sensitivity and interactive features.

The functional scenario state diagram of the smart table lamp, drawn in Lucidchart, is presented in Figure 10.

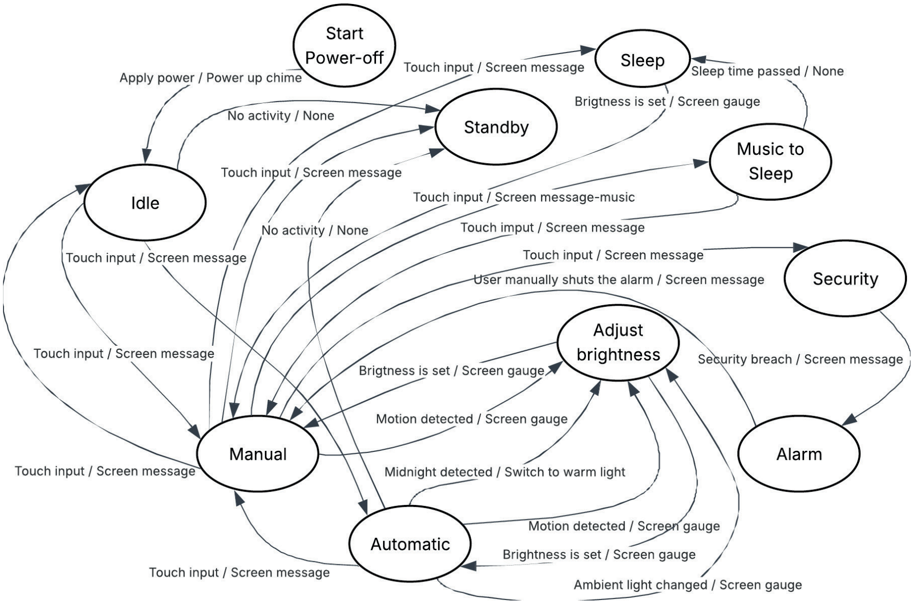


Figure 10: *The state diagram for the functional scenario of the smart table lamp.*

Metrics and measurements:

The planning and development of storyboards, flowcharts, and state diagrams for the design process and the product designed constitute a focused and intensive activity aimed at achieving the design goals of the smart table lamp. Through structured actions and deliberate time investment, we produced a comprehensive set of diagrams representing key aspects of the design workflow and lamp behavior. To evaluate work overall, we tried to get objective "process metrics" by counting micro-actions in the design sessions, as three different types of data:

- **Artifact Metrics** – Counts of things in the diagram itself (Table 7)
- **Interaction Metrics** – Counts of mouse/keyboard actions during creation (Table 8)
- **Time Metrics** – Time passed in the process (Table 9)

Tool	Metric	What It Indicates
Storyboards	Number of frames/sketch panels	Scope of scenarios explored
	Number of annotations per frame	Level of detail in user-context storytelling
	Number of revisions	Degree of iteration/refinement
Flowcharts	Number of process blocks (nodes)	Complexity of procedural logic
	Number of decision diamonds (branches)	How many conditional paths are identified
	Number of loops/feedback connectors	Degree of iterative or error-handling logic
State Diagrams	Number of states	Granularity of behavioral modeling
	Number of transitions	Range of events and responses you're accounting for
	Number of entry/exit actions	Depth of per-state behavior detail

Table 7: *A non-exhaustive list of small "items" that could be logged as artifact metrics*

Event	What It Indicates
Click on the "Add Shape" icon	Frequency of creating new nodes (states/blocks)
Drag-and-drop operations	Effort spent arranging and reorganizing the layout
Text-edit events	How often are labels refined, or notes added
Delete/undo/redo commands	Rate of corrections or back-tracking
Zoom in/out	How often are fine details needed to be inspected?
Pan operations	How much canvas exploration is needed
Switch-tool (pointer ↔ pen)	Mode-switching between drawing vs. selecting

Table 8: *A non-exhaustive list of small "items" that could be logged as interaction metrics*

Metric	What It Indicates
Total time per diagram	Overall effort required
Time per iteration	Speed of converging on a stable design
Idle vs. active time	Periods of thinking vs. drawing

Table 9: *A non-exhaustive list of small "items" that could be logged as time metrics*

Digital tools like Lucidchart or Draw.io can record events directly in their log files or using plug-ins that provide a total count of each item when required. In our case study, the number of diagrams was limited and their sizes manageable, allowing us to count and analyze them manually to collect the research data.

Findings

This section presents a comparative analysis of storyboards, flowcharts, and state diagrams as modeling tools used during the design of a smart product. The analysis is structured around three key dimensions: Artifact, interaction, and time metrics, which are described in the previous section.

All three tools were used to complete the same design task: modeling the behavior and interaction logic of a conceptual smart desk lamp. Metrics are presented in Tables 10, 11, and 12.

Metric	Storyboards	Flowcharts	State Diagrams
Number of frames/blocks/states	24 frames	60 process blocks	23 distinct states
Number of annotations or labels	32 handwritten notes	2 functional labels	23 input/output labels
Number of revisions	3	3	3
Number of decision branches/loops	Not applicable	24 decision diamonds, 4 loops	37 transitions, 3 loop states
Visual structure type	Frame-by-frame scenes	Top-down logic branches	Circular state-event diagram
Visual clarity rating (1–5)	4.5	4	5

Table 10. *Artifact Metrics*

Storyboards emphasized narrative and context, reflected by a high number of frames and handwritten notes. Flowcharts were the most structurally complex, with 60 blocks and multiple decision branches. State diagrams encoded fewer but more interconnected elements, achieving functional compactness. All tools were revised equally (3 iterations), though for different purposes—visual consistency in storyboards, logical path checking in flowcharts, and event-response refinement in state diagrams.

Interaction Event	Storyboards	Flowcharts	State Diagrams
Shape addition (clicks)	30	65	25
Drag-and-drop operations	6	75	27
Text edit actions	40	90	40
Undo/redo operations	8	26	19
Zoom in/out actions	20	32	41
Pan/canvas navigation	30	30	31
Tool switches (draw/select)	56	47	37

Table 11. *Interaction Metrics*

Flowcharts demanded the highest number of user actions, particularly for adding shapes, adjusting layout, and inputting logical details, indicating their labor-intensive and high-control nature. Storyboards required fewer shapes but involved frequent tool switching. State diagrams had a balanced interaction profile, lower total action counts, but high expressive power, suggesting efficient visual encoding of behavior logic.

Time-Related Metric	Storyboards	Flowcharts	State Diagrams
Total time spent (all iterations)	60 min	45 min	35 min
Number of iterations	3	3	3
Average time per iteration	20 min	15 min	11.6 min
Active time vs. idle time	55 / 5 min	35 / 10 min	27 / 8 min
Time efficiency rating (1–5)	2.5	4	4.5

Table 12. *Time Metrics*

Storyboarding was the most time-consuming, primarily due to manual editing and rewriting of the AI-generated images and notes. Flowcharts showed a moderate balance between time and control, while state diagrams required the least time and achieved the highest average modeling efficiency, with faster revisions and fewer idle moments.

Discussion

The findings suggest that storyboards, while effective in visualizing user context and emotional journey, fall short in modeling complex system logic or reactive product behavior. This aligns with prior studies that position storyboards as valuable in the early ideation phase but less suited to representing conditional, algorithmic responses (Greenberg et al., 2012; Rodda et al., 2022). This supports Norman's execution-evaluation cycle, where storyboards represent the user's execution of intent but inadequately reflect the system's evaluation and feedback (Norman, 1986, 1988). In contrast, flowcharts and state diagrams offer more precise modeling capabilities: flowcharts for procedural tasks and logic flows, and state diagrams for dynamic system behavior and context-driven transitions.

The time and interaction data further emphasize how diagrammatic structure influences efficiency. State diagrams required fewer interactions and less time than flowcharts while maintaining modeling accuracy, confirming their suitability for compactly representing product behavior. However, this efficiency comes with a steeper learning curve and less narrative flexibility than storyboards.

From a methodological perspective, this case study was intentionally limited to a single designer, working on a single product. While this scope allowed for in-depth control and introspective analysis, it also imposes limitations on generalizability. The results should not be viewed as statistically representative, but as exploratory insights intended to guide further research. Additionally, the subjective interpretation of diagram clarity, usability, and workflow improvement reflects the author's design expertise and working habits.

The study contributes to a growing body of literature advocating for the integration of logic-based modeling tools into creative design workflows. It highlights the need for tools that bridge the gap between user experience modeling and system-level logic, especially as products become smarter, more responsive, and more connected.

CONCLUSION

This study examined the comparative and complementary use of storyboards, flowcharts, and state diagrams in the context of industrial design, with a specific focus on the planning and modeling processes for smart product development. Through a single-designer case study involving the conceptualization of a smart desk lamp, each tool was tested for its ability to support design process planning, behavior modeling, and interface logic construction.

The findings demonstrate that each diagrammatic tool offers distinct advantages aligned with specific design needs. Storyboards, while visually rich and effective in conveying contextual and narrative sequences, proved the most time-intensive (60 minutes total), with the highest number of tool-switching actions (56) and manual annotations (32). These results reflect their strength in user-centered storytelling, yet also underscore their limitations in adaptability and logic modeling.

Flowcharts emerged as the most interaction-heavy tool, with 65 shape additions, 75 drag-and-drop operations, and 90 text edits across iterations. Their process-driven structure allowed for the clear representation of procedural logic and decision branching, but this came at the cost of increased user workload and planning overhead. On average, each flowchart iteration required 15 minutes of active design work, suggesting a moderate balance between control and effort.

In contrast, state diagrams offered a more compact yet highly expressive way of modeling dynamic system behavior. With only 25 shape additions and a total design time of 35 minutes, they achieved a higher efficiency-to-complexity ratio. Despite fewer user interactions, the diagrams captured 23 distinct states and 37 transitions, demonstrating strong capability in representing reactive behavior and context-sensitive system states.

Beyond the comparative performance of the tools, the study also revealed insights into their potential integration. While flowcharts excel in organizing procedural sequences, and state diagrams clarify event-driven transitions, the two are not mutually exclusive. In practice, each state in a state diagram may contain a procedural loop modeled via a flowchart, and each branch in a flowchart may reference a system state. This suggests that integrated use of both tools, rather than substitution, can support more robust and layered design representations.

Methodologically, the study also underscored the importance of reflective, systematic modeling. The use of empirical metrics—interaction counts, time records, and structural outputs—enabled the evaluation of design efficiency and logic clarity in a transparent and replicable manner. While the single-designer format limits generalizability, the detailed logs and iterative comparisons establish a foundation for future research involving multiple participants or collaborative teams.

Finally, the study contributes to a broader conversation about visual modeling tools in design education and interdisciplinary practice. Given the increasing overlap between design, engineering, and computing in smart product development, diagrammatic tools such as flowcharts and state diagrams offer a shared visual language that can enhance communication, reduce ambiguity, and improve system coherence. Their structured formats help designers move beyond intuition toward more rigorous, testable representations of behavior, interaction, and functionality.

As artificial intelligence becomes increasingly integrated into design environments, flowcharts and state diagrams may serve as foundational frameworks for guiding and interpreting AI-generated outputs. Their structured logic can help designers communicate constraints, anticipate system behaviors, and refine AI-generated concepts through traceable, rule-based visual models. By grounding this research in both practical application and quantitative evaluation, the study positions flowcharts and state diagrams not merely as visualization tools but as active design instruments capable of shaping and clarifying complex product logic.

REFERENCES

Adelt, J., Liebrenz, T., & Herber, P. (2021). Formal Verification of Intelligent Hybrid Systems that are Modeled with Simulink and the Reinforcement Learning Toolbox. In M. Huisman, C. Pasareanu, & N. Zhan (Eds.), Formal Methods (pp. 349–366). Springer International Publishing. https://doi.org/10.1007/978-3-030-90870-6_19

Alhir, S. S. (2003). Learning UML. O'Reilly Media, Inc.

Alhumaidan, F. (2012). State Based Static and Dynamic Formal Analysis of UML State Diagrams. Journal of Software Engineering and Applications, 5(7), Article 7. https://doi.org/10.4236/jsea.2012.57056

Bagnati, E., & Del Bello, A. (2024). Modular UML statechart modelling for simulating the behaviour of production systems with integrated dispatching policies. https://www.politesi.polimi.it/handle/10589/230526

Basheri, M. (2010). Collaborative Learning of UML-State diagrams using Multi-Touch Technology. Durham University.

Bowman, A. (2024). Intelligent Student Learning System (iSLS) [M.S.]. https://www.proquest.com/docview/3093581801/abstract/5AFEA474A8464340PQ/1

Campbell, L. H., & McDonagh, D. (2009). Visual Narrative Research Methods as Performance in Industrial Design Education. Qualitative Inquiry, 15(3), 587–606. https://doi.org/10.1177/1077800408318306

Chinofunga, M. D., Chigeza, P., & Taylor, S. (2025). How can procedural flowcharts support the development of mathematics problem-solving skills? Mathematics Education Research Journal, 37(1), 85–123. https://doi.org/10.1007/s13394-024-00483-3

Creately. (2025). Horizontal Flowchart. https://creately.com/diagram/example/jny57xed8/horizontal-flowchart-template

Diagramming Powered By Intelligence. (2025). Lucidchart. https://www.lucidchart.com/pages/landing

Doyle, P. (2002). Believability through context using "knowledge in the world" to create intelligent characters. Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 1, 342–349. https://doi.org/10.1145/544741.544822

Draw.io. (2025). https://app.diagrams.net/

Garzon, S. R., & Louis, B. (2020). Context Flow Graphs: Situation Modeling for Rule-Based Proactive Context-Aware Systems. IEEE Journals & Magazine, IEEE Access, 8, 212939–212960. https://doi.org/10.1109/ACCESS.2020.3040060

Giuliani, F., Frizziero, L., & Donnici, G. (2024). Industrial Design Structure Plus: Industrial Design Structure Method Implemented with Theory of Inventive Problem Solving and Conceptual Method. https://cris.unibo.it/handle/11585/1006529

Goldberg, R. P. (1973). Architecture of virtual machines. Proceedings of the Workshop on Virtual Computer Systems, 74–112. https://doi.org/10.1145/800122.803950

Goldman, D. B., Curless, B., Salesin, D., & Seitz, S. M. (2006). Schematic storyboarding for video visualization and editing | ACM Transactions on Graphics. ACM Transactions on Graphics, 25(3), 862.

Granfelt, S. A. (2017). A Flowchart Guide For Micro & Small Business [BSc., Arcada]. https://www.theseus.fi/bitstream/handle/10024/125035/Stephan%20Aschwanden-Granfelt.pdf?sequence

Greenberg, S., Carpendale, S., Marquardt, N., & Buxton, B. (2012). The narrative storyboard: Telling a story about use and context over time. Interactions, 19(1), 64–69. https://doi.org/10.1145/2065327.2065340

Guiochet, J. (2015). Trusting robots: Contributions to dependable autonomous collaborative robotic systems [Thesis, Université de Toulouse 3 Paul Sabatier]. https://hal.science/tel-01276555

Harel, D. (1987). Statecharts: A visual formalism for complex systems. Science of Computer Programming, 8(3), 231–274. https://doi.org/10.1016/0167-6423(87)90035-9

Helin, K., Evilä, T., Viitaniemi, J., Aromaa, S., Kilpeläinen, P., Rannanjärvi, L., Vähä, P., Kujala, T., Pakkanen, T., Raisamo, R., Salmenperä, P., Miettinen, J., & Patel, H. (2007). HumanICT. New Human-Centred Design Method and Virtual Environments in the Design of Vehicular Working Machine Interfaces. VTT Working Papers, 1459(7683). http://www.vtt.fi/publications/index.jsp

Herman, G. L., & Choi, D. S. (2017). The Affordances and Constraints of Diagrams on Students' Reasoning about State Machines. Proceedings of the 2017 ACM Conference on International Computing Education Research, 173–181. https://doi.org/10.1145/3105726.3106172

Hossain, Q., Yasmin, F., Biswas, T. R., & Asha, N. B. (2024). Integration of Big Data Analytics in Management Information Systems for Business Intelligence. Saudi Journal of Business and Management Studies, 9(09), 192–203. https://doi.org/10.36348/sjbms.2024.v09i09.002

Jacob. (1985). A State Transition Diagram Language for Visual Programming. Computer, 18(8), 51–59. https://doi.org/10.1109/mc.1985.1662976

Jäger, D., Schleicher, A., & Westfechtel, B. (1999). Using UML for software process modeling. SIGSOFT Softw. Eng. Notes, 24(6), 91–108. https://doi.org/10.1145/318774.318788

Jonassen, D. H. (2000). Toward a design theory of problem solving. Educational Technology Research and Development, 48(4), 63–85. https://doi.org/10.1007/BF02300500

Kant, E. (1985). Understanding and Automating Algorithm Design. IEEE Transactions on Software Engineering, SE-11(11), 1361–1374. https://doi.org/10.1109/TSE.1985.231884

Kim, K., & Lee, K. (2016). Collaborative product design processes of industrial design and engineering design in consumer product companies. Design Studies, 46, 226–260. https://doi.org/10.1016/j.destud.2016.06.003

Konrad, S., & Cheng, B. H. C. (2002). Requirements patterns for embedded systems. Proceedings IEEE Joint International Conference on Requirements Engineering, 127–136. https://doi.org/10.1109/ICRE.2002.1211541

Koskimies, K., & Mäkinen, E. (1994). Automatic synthesis of state machines from trace diagrams. Software: Practice and Experience, 24(7), 643–658. https://doi.org/10.1002/spe.4380240704

Lucidchart. (2025). Lucidchart. https://www.lucidchart.com/pages/landing

Ma, J., & Yu, S. (1998). Practical rules for reduction on the number of states of a state diagram. Proceedings. Technology of Object-Oriented Languages. TOOLS 26 (Cat. No.98EX176), 46–55. https://doi.org/10.1109/TOOLS.1998.711002

Micheli, P., Jaina, J., Goffin, K., Lemke, F., & Verganti, R. (2012). Perceptions of Industrial Design: The “Means” and the “Ends.” Journal of Product Innovation Management, 29(5), 687–704. https://doi.org/10.1111/j.1540-5885.2012.00937.x

Millhouse, T. (2018). Virtual Machines and Real Implementations. Minds and Machines, 28(3), 465–489. https://doi.org/10.1007/s11023-018-9472-7

Newman, R. M. (1998). A VISUAL DESIGN METHOD AND ITS APPLICATION TO HIGH RELIABILITY HYPERMEDIA SYSTEMS [PhD. Thesis, University of Plymouth.]. https://pearl.plymouth.ac.uk/fose-theses-other/355

Norman, D. (1986). User-Centered System Design: New Perspectives on Human-Computer Interaction. Lawrence Erlbaum Associates.

Norman, D. (1988). The psychology of everyday things (pp. xi, 257). Basic Books.

Olivé, A. (Ed.). (2007). State Transition Diagrams. In Conceptual Modeling of Information Systems (pp. 299–323). Springer. https://doi.org/10.1007/978-3-540-39390-0_13

Östlund, A. (2022). Telling a story of the future: Using storyboards and narratives to evaluate anticipated experiences. https://urn.kb.se/resolve?urn=urn:nbn:se:sh:diva-49410

Palovuori, J. (2017). Towards harmonious experiences – A service design approach to evaluating and optimizing multi-touchpoint user experience [MSc.]. https://aaltodoc.aalto.fi/handle/123456789/27146

Pavel, M. (1978). MICROPROCESSORS IN PSYCHOLOGY: Behavior Research Methods & Instrumentation, 10(2), 238–240.

R. Ponelis, S. (2015). Using Interpretive Qualitative Case Studies for Exploratory Research in Doctoral Studies: A Case of Information Systems Research in Small and Medium Enterprises. International Journal of Doctoral Studies, 10, 535–550. https://doi.org/10.28945/2339

Rodda, J., Ranscombe, C., & Kuys, B. (2022). A method to explore strategies to communicate user experience through storyboards: An automotive design case study. AI EDAM, 36, e16. https://doi.org/10.1017/S0890060421000287

Self, J. A. (2012). The Use of Design Tools in Industrial Design Practice [PhD. Thesis, Kingston University]. https://www.researchgate.net/publication/289674998_The_Use_of_Design_Tools_in_Industrial_Design_Practice

Shaw, A. C. (1992). Communicating Real-Time State Machines—ProQuest. IEEE Transactions on Software Engineering, 18(9), 805–816.

Shina, S. G. (2012). Concurrent Engineering and Design for Manufacture of Electronics Products. Springer Science & Business Media.

Simon, M. (2012). Storyboards: Motion In Art (3rd ed.). Routledge. https://doi.org/10.4324/9780080465951

Spencer, V. (2025, April 19). Storyboarding for Product Design. Instructables. https://www.instructables.com/Storyboarding-for-Product-Design/

State diagram. (2025). In Wikipedia. https://en.wikipedia.org/w/index.php?title=State_diagram&oldid=1282429372

Truong, K. N., Hayes, G. R., & Abowd, G. D. (2006). Storyboarding: An empirical determination of best practices and effective guidelines. Proceedings of the 6th Conference on Designing Interactive Systems, 12–21. https://doi.org/10.1145/1142405.1142410

Vegte, W. F., & Breemen, E. J. J. (2009). Flowchart-Assisted Function Analysis of Products to Support Teaching of the Exact Sciences. DS 58-10: Proceedings of ICED 09, the 17th International Conference on Engineering Design, Vol. 10, Design Education and Lifelong Learning, Palo Alto, CA, USA, 24.-27.08.2009, 101–112.

Vizcom AI. (2025, February). Online Artificial Intelligence Visualization Services. https://www.vizcom.ai/

Vpadmin. (2023, March 22). Introduction to State Diagrams: A Comprehensive Guide for Software Engineering. Visual Paradigm Guides. https://guides.visual-paradigm.com/introduction-to-state-diagrams-a-comprehensive-guide-for-software-engineering/

Wah, B. W., Ieumwananonthachai, A., Yao, S., & Yu, T. (1995). Statistical generalization: Theory and applications. Proceedings of ICCD '95 International Conference on Computer Design. VLSI in Computers and Processors, 4–10. https://doi.org/10.1109/ICCD.1995.528783

Wang, J., Ranscombe, C., & and Eisenbart, B. (2024). An integrated prototyping tool to enhance interdisciplinary communication in smart product design. Ergonomics, 1–19. https://doi.org/10.1080/00140139.2024.2418949

Wang, J., Ranscombe, C., & Eisenbart, B. (2023). Prototyping in smart product design: Investigating prototyping tools to support communication in the early stage smart product development. International Journal of Design Creativity and Innovation, 11(3), 159–184.

Welte, T. M. (2009). Using State Diagrams for Modeling Maintenance of Deteriorating Systems. IEEE Transactions on Power Systems, 24(1), 58–66. https://doi.org/10.1109/TPWRS.2008.2005711

Weng, W. (2024). Flowchart. In W. Weng (Ed.), A Beginner's Guide to Informatics and Artificial Intelligence: A Short Course Towards Practical Problem Solving (pp. 13–20). Springer Nature. https://doi.org/10.1007/978-981-97-1477-3_2

Willett, D. L., Pandey, A., Ifejika, Nn. L., Kannan, V., Berry, J. D., & Basit, M. A. (2018). State Diagrams for Automating Disease "Risk Pyramid" Data Collection and Tailored Clinical Decision Support. Proceedings of the 2018 ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics, 551–552. https://doi.org/10.1145/3233547.3233660

Wynn, D. C., & Eckert, C. M. (2017). Perspectives on iteration in design and development. Research in Engineering Design, 28(2), 153–184. https://doi.org/10.1007/s00163-016-0226-3

GÖRSEL KAYNAKLAR:

Figure 1: Spencer, V. (2025, April 19). Storyboarding for Product Design. Instructables. https://www.instructables.com/Storyboarding-for-Product-Design/

Figure 2: Creately. (2025). Horizontal Flowchart. https://creately.com/diagram/example/jny57xed8/horizontal-flowchart-template

Figure 3: Left-State diagram. (2025). In Wikipedia. https://en.wikipedia.org/w/index.php?title=State_diagram&oldid=1282429372

Right- Author's work

Figure 4: Vpadmin. (2023, March 22). Introduction to State Diagrams: A Comprehensive Guide for Software Engineering. Visual Paradigm Guides. https://guides.visual-paradigm.com/introduction-to-state-diagrams-a-comprehensive-guide-for-software-engineering/

Figure 5: Author's drawings

Figure 6: Author's drawings

Figure 7: Author's drawings

Figure 8: Author's drawings

Figure 9: Author's drawings

Figure 10: Author's drawings

INTERNET KAYNAKLARI

Creately. (2025). Horizontal Flowchart. https://creately.com/diagram/example/jny57xed8/horizontal-flowchart-template

Diagramming Powered By Intelligence. (2025). Lucidchart. https://www.lucidchart.com/pages/landing

Draw.io. (2025). https://app.diagrams.net/

Lucidchart. (2025). Lucidchart. https://www.lucidchart.com/pages/landing

Spencer, V. (2025, April 19). Storyboarding for Product Design. Instructables. https://www.instructables.com/Storyboarding-for-Product-Design/

State diagram. (2025). In Wikipedia. https://en.wikipedia.org/w/index.php?title=State_diagram&oldid=1282429372

Vizcom AI. (2025, February). Online Artificial Intelligence Visualization Services. https://www.vizcom.ai/

Vpadmin. (2023, March 22). Introduction to State Diagrams: A Comprehensive Guide for Software Engineering. Visual Paradigm Guides. https://guides.visual-paradigm.com/introduction-to-state-diagrams-a-comprehensive-guide-for-software-engineering/