Background On Spatial Computing

Historical Context On Operating System

To meaningfully appreciate the application and utility of spatial computing, it is critical to provide the context required to familiarize ourselves with the concepts at play and the root of the computing paradigm.

An Operating System [OS] is an abstract notion within computing; it is defined as the software interface between a user and the computer hardware. An OS manages, protects, and coordinates computer resources to facilitate tasks for the user in the software environment. The need for an OS arose from the requirement to manage computer hardware resources efficiently (CPU, memory, I/O devices) and to provide a user-friendly medium for the user to interact with the computer system ("Components of Operating System," n.d.).

As computer capabilities and user requirements evolved rapidly through the late 20th century, OSs adapted to support computer paradigms that were enabled by novel hardware tools, including multithreading, file systems, multi-users, and GUIs (Dhanjal, 2022).

Today, OSs are utilized in modern applications that expand the original interface scope of the OS; it has evolved to support distributed computing, cloud integration, and virtual simulations while maintaining user security. With well-defined modularity, the Kernel is the core of the OS, managing resources and hardware/software interfaces, with the shell being the platform for user interaction.

The key functions, more specifically, of the OS can be seen as the following (Dhanjal, 2022; "Components of Operating System," n.d.):

1. Process Management: Create, schedule, manage, and terminate processes/threads running within a system. The OS can benefit from parallel processing and can use the scheduler to optimize for CPU utilization and throughput.
2. Memory Management: Allocates and manages the system's main memory, benefits from disk memory and caching paradigms. Virtual memory is a core component of memory management that extends the physical memory onto disk storage and simplifies management for users.
3. File System Management: Manages data storage, allowing data to be stored, retrieved, and organized by leveraging the disk and abstract structures similar to memory management.

a. **Data Blocks**: Basic units of storage in a file system. A data block is a contiguous set of bytes on a disk. Files are stored as a series of data blocks that need not be ordered.
b. **iNode (Index Node)**: In file systems, an iNode is a data structure that represents a file. It stores metadata about a file and the disk block locations where the file's contents are stored. The amount of store capabilities of an Inode between direct, single, secondary and tertiary pointers depends on the block size and pointer size.
c. **FDT (File Descriptor Table)**: When a file is opened, the operating system creates an entry in the FDT, which includes information such as the current position in the file, the file mode, and a pointer to the file's iNode. This is cached locally by OS and is not stored on disk.
d. **Caches**: In the context of file systems, caches are used to improve performance. Data that is accessed frequently is stored in a cache, which is a faster type of memory. This can be an iNode, Directory and FBM cache; it reduces the need to read data from or write data from disk.
e. **FBM (Free Block Management)**: This is an abstract type of data storage used to track which blocks on a disk are free and which are in use. It's essential for efficiently allocating space for new files and for files that are growing, ensuring that the file system can find free blocks quickly.

4. I/O Management: Facilitating user interaction and data exchange by leveraging different drivers/controllers; further extends to communication between devices and kernels.
5. Networking: Remote and local procedure calls and larger network connections

Spatial Computing Overview

Spatial Computing is the frontier of modern computing architecture. In an accelerated environment of hardware and software development, the world is evolving to be interdimensional, and overlaying software and physical environments is critical to moving forward. Spatial Computing is leveraged to synthesize, analyze, and interact with the physical world through a software-enabled environment (The Manufacturing Leadership Council, 2021). '

Spatial Computing traces to early VR work with sensors in the 1960s and has now evolved to sophisticated system architecture integrating 3D modelling, real-time data acquisition/processing, and multifaceted sensor technologies.

Spatial Computing Definition

As an extension of basic computing, spatial computing introduces a new dimension to computing: physical space. We can view spatial computing at its core as the mechanics to process elements both with a memory and physical space context to execute a program, whereas traditional computing interacts purely within a memory context and available data (PTC, 2023).

Spatial computing is not just about visualizing computing in space or mapping physical space. It is additionally about integrating physical space as a core component of the computational process. This creates a duality relationship where physical context can affect memory, and memory context can affect spatial understanding.

Types of Spatial Computing & Needs

Spatial Computing has distinct permutations with unique use cases: Augmented Reality (AR) migrates digital information into the physical world, Virtual Reality (VR) develops an engulfed virtual landscape for the user, Mixed Reality (MR) as a blend of AR and VR, and Sensor Network which allows sensor architecture to collect data on the physical world for tangible applications such as Robotics and Autonomous Vehicles (Markovate, 2023).

As Spatial Computing has a wide array of defined frameworks, the current applications vary as well, with modern applications in industries including Gaming [VR], Healthcare [MR], Manufacturing, Education and Defense, to name a few.

Modern Hardware/Software Requirments for Spatial Systems

Similar to modern Operating Systems, Spatial Computing architectures are incredibly nuanced and modular systems, relying on synchronously functioning software and hardware elements. At their base, Spatial Computing systems require sensors [input devices] to synthesize and capture real-world spatial data, processors [~advanced CPU] to compute and process data, an output device [application sensitive] to display the computed information, and an integration of strong software frameworks. These can include simulation machines, computer vision software,

modular AI packages, and cloud computing platforms that facilitate the advanced requirements and resource-intensive urgency of spatial computing. Similar to the general OS structure, this can be distilled down to a few essential functions that Spatial Computing systems must maintain, including data processing, data capture, user interface, and spatial mapping/navigation.

High-level Advantages/Disadvantages of Spatial Computing

The rapidly emerging ability of Spatial Computing will lead to utilities beyond those offered by relic system architectures such as traditional computer/phone OS (PCMAG, n.d.).

Spatial Computing will lead to augmented interactivity and interoperability between the physical and software realms, leading to immersive user experiences and enhanced cognitive decision-making frameworks [e.g., EVs].

However, as with any innovative technology, rapid growth can lead to oversights that may have malignant impacts on users and system efficacy. Spatial data collection and storage can lead to a plethora of immediate technical concerns, including high compute costs, efficient storage, data security, and user-centric concerns, including privacy issues due to the rapid capture of spatial data and limited access to technology due to the current and foreseeable high costs of acquiring the necessary hardware.

Comparison With Traditional Computer Systems

Spatial Computing is a significant shift from traditional operating systems that focused predominantly on the software domain. As spatial systems require a coherent and continuous understanding of their surroundings, the system necessitates more complex hardware integration [sensors, render] than traditional input/output devices [keyboard, mouse], and the software stack [3D Data Capture/Process/Storage and Enhanced UI].

The real-world elements require the systems to evolve from a traditional OS and adapt to its advanced computing needs. While both types of systems do need to manage resources like memory and computing, spatial Computing requires a more contextualized view of data and can have higher computation resource requirements.

Industry & Research Frontier

Due to Spatial Computing's vast applications, there is substantial interest in the industry, with leading incumbents building out software and hardware for consumers. Research Labs are maintaining frontier technology testing and building efficiency for existing functions.

Companies like Google, Microsoft, and Apple are leading the consumer effort, launching innovative software-enabled AR experiences with different wearable technologies that enhance user experience with spatial data. On the other hand, hardware players like NVIDIA, Intel, and Qualcomm are offering support for efficient storage and computing capabilities (Hackl, 2023).

Spatial Computing, being the future of how we compute and now ingraining spatial data, is the next generation of computing and hardware. Research Labs renowned for novel experiences are leading the frontier of work, including MIT Media Lab, Stanford University, and Carnegie Mellon University. The biggest challenges in Spatial Computing are spatial mapping, resource allocation among hardware, and efficient computing.

Apple Leading Spatial Computing

Apple is an incumbent in the technology industry and is now navigating the frontier of the spatial computing ecosystem.

As Apple has a broad breadth and depth in the technological industry, it works with a keen focus on integrating spatial-centric tech into its larger existing product ecosystem. This leads to an understanding that any spatial computing technology they develop is designed to work seamlessly with other Apple products, which are firmly consumer-centric. In spatial computing, this could translate to intuitive interfaces and AR interfaces. Apple has actually built a framework for AR development known as ARKit to build AR experiences within iOS devices. On the hardware front, Apple has been leading the edge with their now-launched Vision Pro headset, which, unlike competitors, they have branded as spatial computing.

They lead this by building out VisionOS, the first spatial operating system that blends the physical and digital worlds. Apple's spatial computing also involves the integration of advanced sensors and cameras to capture comprehensive visual data and map to the VisionOS three-dimensional interface for users (Boreham, 2023).

Future of Spatial Computing

Spatial Computing is the future of next-gen computer architecture as our ability to build effective solutions with frontier tech expands. Spatial Computing projects like JAMScript can facilitate the extension of this architecture to enable the tangible applications of projects such as Smart Cities, IoT, and ubiquitous computing [fog, cloud, worker architecture]. With the rapidly evolving capabilities of ML as an additional instrument, real-time predictive engines will additionally become more efficient and accurate.

SIB Structure [S-Block, T-Node, Trails] (From Notion)

As we understand from the structure and utility of Spatial Computing, the core functions required are data collection and data storage [structured format for geo-centric data] and data querying.

File System Comparison

The SIB system is inspired by the abstraction of a File System for the OS. In a file system, data is stored in data blocks on the disk. iNodes are used as a data structure to keep information about a file and the data blocks storing its content. The backbone of the storage is a physical device such as a disk and drives you'd find on a computer.

S-Block

SIBs are additionally placed as an adaptation of File Systems for Spatial Computing. In an SIB system, S-Blocks represent units of spatial data; each block corresponds to a specific area in the physical world, similar to a data block corresponding to a segment of byte data on the disk. A given physical space is divided into sub-spaces called S-Blocks which can occur in 2D or 3D space. We transform the subspaces through an S-Block Numbering System to map the physical grid into software. As in real-time 'within' an S-Block, there can be real nodes active, and those devices can publish messages to observers within the same block. Unlike data blocks, S-Blocks are active and dynamic; they can have events and nodes travel in and out. S-Blocks enable the representation of real-time data with dynamic changes. Data blocks are static and only change if data is modified, whereas S-Blocks are continuously updatable. S-Blocks are further curated to represent contextual spatial information of a mapped geo plane and their real-time events,

movements. Moreover, this provides support for real-time analytics done by edges to reflect the nature of spatial events.

Trails

Trails are a sequence of S-Blocks that group the node's interaction within the physical space; like a file path, a trail showcases a way to trace and access specific sequences within the spatial data. Trails are, at their core, a log of a node's path over a space, and they are incredibly powerful for identifying future movements and conducting analytics. Trails specifically 'trace' a sequence of S-Blocks and their data, whereas file paths are a location of data in a hierarchical file system. Trails allow for the storage of dynamic sequential data on a node's path and movement across S-Blocks. This can be incredibly powerful for aiding in decision-making processes and efficiency frameworks for nodes, e.g. applications for autonomous vehicles.

T-Node

T-Nodes are comparable to iNodes. They store information on sequences of S-Blocks associated with a node and its trail, as well as the relevant contextual metadata regarding start-time, length, and time in S-Block. T-Nodes allow the linking of temporal and spatial data that can be used more effectively for predictive algorithms on nodes. T-Nodes allow a hierarchical structure for organizing and storing spatial data regarding a node and further enable efficient access and data queries. T-Nodes allow for a holistic scope on spatial data of a node and are the backbone of an SIB system.

Requirements

This requires an incredibly vast storage for data; due to the high volume and cost/feasibility of hardware, a server/cloud-based architecture is required to maintain the spatial data. The data in SIB is accessed through APIs or operations like querying specific S-Blocks or analyzing trails within T-Nodes by reaching the storage server/cloud. As conjecture, this may require more than a single server, beyond a general MQTT server. As the systematic approach with the SIB is distinct from the File System, the required hardware components also change. Sensors are essential to actually collect real-time spatial data, and server/sound infrastructure allows for storage and scalable computing at an expense.

JAMScript/Jamvis Analysis (from Notion)

JAMScript

JAMScript is a polyglot language built to be a framework for cloud architecture in IoT-enabled cities and other applications. There are controllers [at a high cloud, fog level] and workers [compute]. They can communicate through remote calls of flows and flows to exchange meaningful information, and at the highest level, information gets reverted back to the cloud infrastructure.

JAMScript provides a prominent enhancement from active distribution compute standards by leveraging evolved modern systems. It offers efficient resource utilization across edge devices, enhanced data handling, and low latency.

JAMScript architecture provides two key role types for nodes: workers and controllers.

Worker Nodes execute tasks and handle data. They are generally lower-level edge nodes of a network closer to sources, allowing efficient computing.

Controller nodes manage distribution and communication for workers and relay information to cloud architecture for broad storage.

This architecture is key in distributed systems like IoT networks, where managing vast numbers of devices efficiently is crucial. JAMScript involves controller-to-worker (downward) and worker-to-controller (upward) calls.

JAMScript Regarding Spatial Computing

JAMScript, when considered with spatial computing, is designed to focus on nodes and their requirements/functions with spatial domains.

Nodes in JAMScript exist within a spatial domain, which can include physical and abstract [software] spaces, and nodes themselves can be stationary or mobile. Allowing for a myriad of applications that can be built utilizing these core tenets of JAMScript.

This does raise challenges regarding node discovery and management, which requires a firm view and management of spatial node data stored in T-Nodes and S-Blocks. As nodes are dynamic, the system requires that is not just reactive to changes but also proactive, anticipating events and adapting accordingly, which can be done by utilizing trails.

Jamvis

Jamvis is a tool designed to visualize the JAMScript simulation. Jamvis allows for a visual understanding of each node and its state. This can change dynamically as the node state changes and users/other nodes interact with it.

Within Jamvis mapping, the physical space is mapped into a coordinate space, broken into cells, which are S-Blocks that showcase a node's geo-coordinates that map to a real-time location of the node.

As we mentioned earlier, S-Blocks are the building blocks that discretize and define the physical space.

To build the true spatial computing architecture, launch an S-Block server, initialize the nodes within the JAMScript program, and facilitate the visualization. The grid, now with S-Blocks, can showcase the location for any given node in the program whose contexts are provided through MQTT topics [structured strings used in the MQTT protocol to route messages between nodes].

Concerns/Questions Brodadly + Solutions For Spatial Compute, SIB, JAMScript/Jamvis [Concepts]

Accuracy and Precision of S-Block Numbering

To prevent spatial data errors, have precise conversion from GPS or other coordinate systems to S-Block numbering.

Solution: For either 2D or 3D mapping, it will be essential to have a robust mapping algorithm from the original world-view/GPS-coordinate system to S-Block conversion. Simple ways to help build structure are by making S-Blocks of the same size and breaking them into grids similar to rows + columns.

Message Publishing and Handling

Efficiently managing message publishing within S-Blocks, especially in areas with high device density and message volume.

Solution: Build efficient message compression techniques to pass a high message volume. Message queuing systems or ‘mail’ systems can also help distribute messaging.

MQTT Server Scalability and Distribution

Use scalable cloud methodology to expand server requirements. Implement a load-balancing mechanism to distribute traffic across multiple MQTT servers. Based on the storage requirement, segregate MQTT servers by style/type of node.

Structure and Management of T-Nodes

Efficiently designing T-nodes to handle S-Block pointers and time trails, adapting to increasing data volume and complexity.

Solution: T-Nodes should be an efficient data structure built to reduce overhead and optimize querying as they maintain node spatial data. Implement caching mechanisms to reduce database load for 'live nodes.'

Garbage Collection and Data Lifespan

Implementing effective garbage collection algorithms for trails, balancing data relevance with storage optimization.

Solution: As nodes finish paths or data become irrelevant, building a data garbage collection algorithm similar to memory management and 'page-free' mechanics is essential. Opportunity to utilize ML to scale the solution to be more efficient as utilized.

Distributed System Resilience

As spatial data needs, S-Nodes can be dynamic; it is pragmatic to build an architecture that can maintain system availability during network disconnections or disruptions.

Solution: Build a rigorous data backup and access plan and use decentralized architectures that can cache and store data to enforce resilience.

Node Discovery and Mobility

Implementing discovery processes for both stationary and mobile nodes can be challenging; however, there are many real-time benefits with tangible applications [EVs + Safety].

Solution: Utilize real-time data analytics to improve discovery accuracy for adjacent nodes; cached data can make this more efficient. Predict node movement from trails to discover the future position of nodes in the general system and distribute information across the architecture.

Dynamic S-Block Management

Allowing for dynamic adjustment of S-Block sizes and effectively managing these changes. Based on mapping, it will be more effective for spatial utilization to adjust the S-Block sizes through relevant historical data and algorithms. It is critical to optimize mapping and S-Block management as S-Blocks are memory-intensive ADTs.

S-Block Server Integration and Management

Integrating SBS into the system effectively, ensuring proper management of S-Block relationships.

Solution: Build clear, firm protocols for the framework of S-Block management within the server.

Distributed Architecture Implementation

Creating a fully distributed system without central components, capable of functioning in both connected and ad-hoc networks.

Solution: Potentially leverage NS3 protocols for decentralized network protocols to reduce reliance on central components.

Additional Questions

1. How will the system adapt to dynamic changes within S-Blocks, such as moving objects or environmental conditions?

Solution: We can implement real-time monitor architecture within JAMScript to track active node changes. Within SIB we can add mechanics for messaging updates on S-Block data/struct.

2. How will the system be adapted for different spatial environments, from open outdoor spaces to complex indoor settings?

Solution: Design S-Blocks with flexibility to adapt to various spatial environments. Have them be physical-space centric rather than what is tangibly in a given block; this way, it can be location agnostic and focus on the area [at least for 2D].

3. How will real-time message delivery be ensured in high-density S-Blocks without compromising performance?

Solution: The potential solution includes relying on local cache and edge computing to have efficient messaging protocols to view systems in S-Blocks. Opportunity for focus resources on high-density areas, allowing for more efficient resource deployment. If the goal is efficiency on throughput, we can sacrifice space in local or cloud storage.

4. What mechanisms exist for efficiently retrieving and querying of trail data in time-critical applications?

Solution: Build a push-based notification system that triggers a constant update of trails within T-Nodes. On the hardware and software integration front, key motion sensor stacks and GPS mechanics will be needed to update information on trails as well.

We can implement caching strategies to improve response times for trail data retrieval.

5. How will the discovery process dynamically adapt to the mobility and changing proximities of nodes?

Solution: Implement geo-proximity techniques within nodes traveling within a program to detect adjacent nodes locally and send updates to the larger SIB system. S-Nodes can have updated protocols to message the SIB/nodes about nodes within proximity [based on pre-set conditions].

Spatial Computing Comparison with Apple

Apple, as a consumer enterprise, prioritizes user experience simplicity and ecosystem integration in its approach to spatial computing. Their technical stack, including ARKit and LiDAR scanners, is designed to integrate virtual elements with the physical world on iOS devices (Evans & Evans, 2024). Furthermore, Apple places a strong emphasis on consumer-data privacy, influencing their approach to spatial computing within larger node architectures.

In contrast, our architecture, integrating JAMScript and SIB, is more focused on data management and tracking, aligning closely with IoT and industrial applications. This approach differs from Apple's, which is more geared towards enhancing user interaction in everyday

applications. Our architecture utilizes S-Blocks to discretize physical space and Trails and T-Nodes for logging and storage in spatial domains, making it suitable for a broader range of applications in industrial, scientific, or urban planning contexts. While Apple focuses on hardware, tailoring their spatial computing to sensor technology, our architecture emphasizes node-based solutions (Ghosh & Singh, 2023).

Citation

1. Boreham, J. (2023, November 2). Spatial computing 101: Everything you need to know. The Metaverse Insider. Retrieved from https://metaverseinsider.tech/2023/10/23/spatial-computing-101-everything-you-need-to-know/
2. Dhanjal, S. (2022, July 7). Components of operating system. Scaler Topics. Retrieved from https://www.scaler.com/topics/components-of-operating-system/
3. Evans, A. H. B. J., & Evans, J. (2024, January 9). How to think about Apple and spatial computing. Computerworld. Retrieved from https://www.computerworld.com/article/3712160/how-to-think-about-apple-and-spatial-computing.html
4. Ghosh, D., & Singh, A. (2023, November 2). Exploring vision OS: The future of Augmented Reality on mobile. The Talent500 Blog. Retrieved from https://talent500.co/blog/exploring-vision-os/
5. Hackl, C. (2023, November 30). Ai is a critical building block for spatial computing. Forbes. Retrieved from https://www.forbes.com/sites/cathyhackl/2023/11/29/ai-is-a-critical-building-block-for-spatial-computing/
6. Markovate. (2023, October 13). What is spatial computing? How it will impact your business? Retrieved from https://markovate.com/blog/spatial-computing/
7. PCMAG. (n.d.). What is spatial computing? A basic explainer. Retrieved from https://www.pcmag.com/how-to/what-is-spatial-computing-a-basic-explainer
8. PTC. (2023, November 10). What is spatial computing: Industry insights. Retrieved from https://www.ptc.com/en/industry-insights/spatial-computing
9. The Manufacturing Leadership Council. (2021, September 29). Spatial computing: Digitizing the future of work. Retrieved from https://manufacturingleadershipcouncil.com/spatial-computing-digitizing-the-future-of-work-17498/
10. Components of operating system. Tutorialspoint. (n.d.). Retrieved from https://www.tutorialspoint.com/operating_system/os_components.htm

Visuals

Spatial Mapping

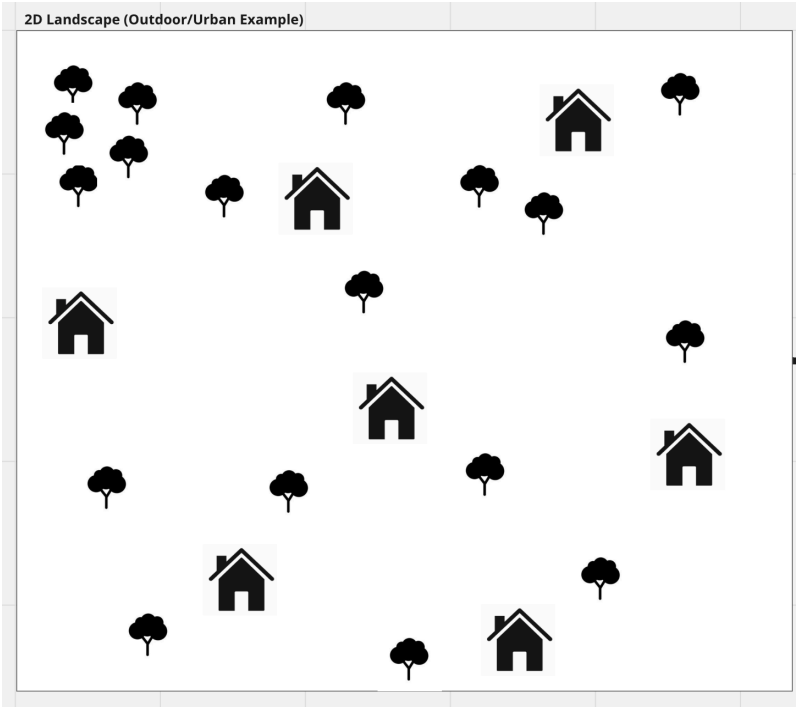

Figure 1: Environment of Program

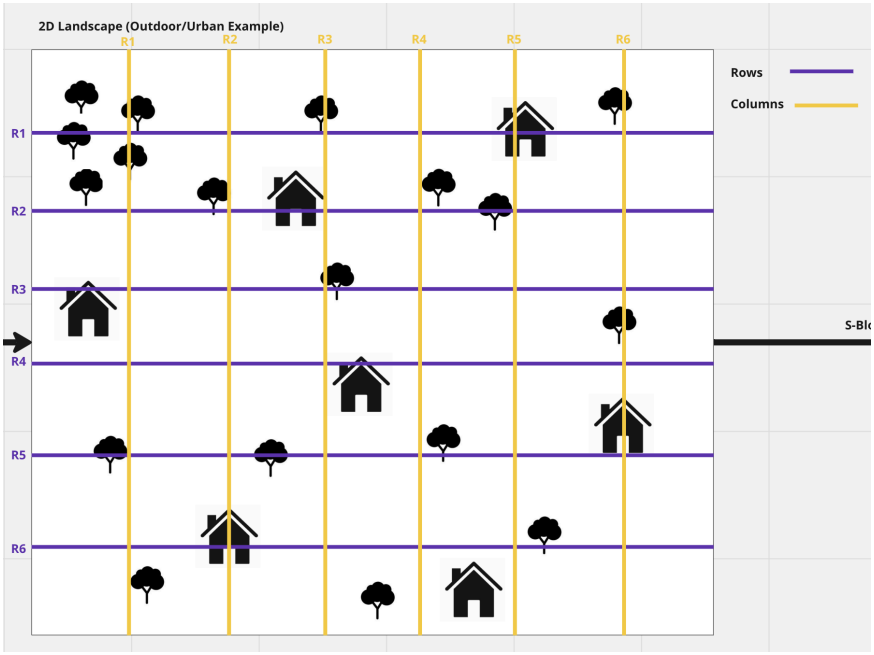

Figure 2: Spatial Mapped Environment

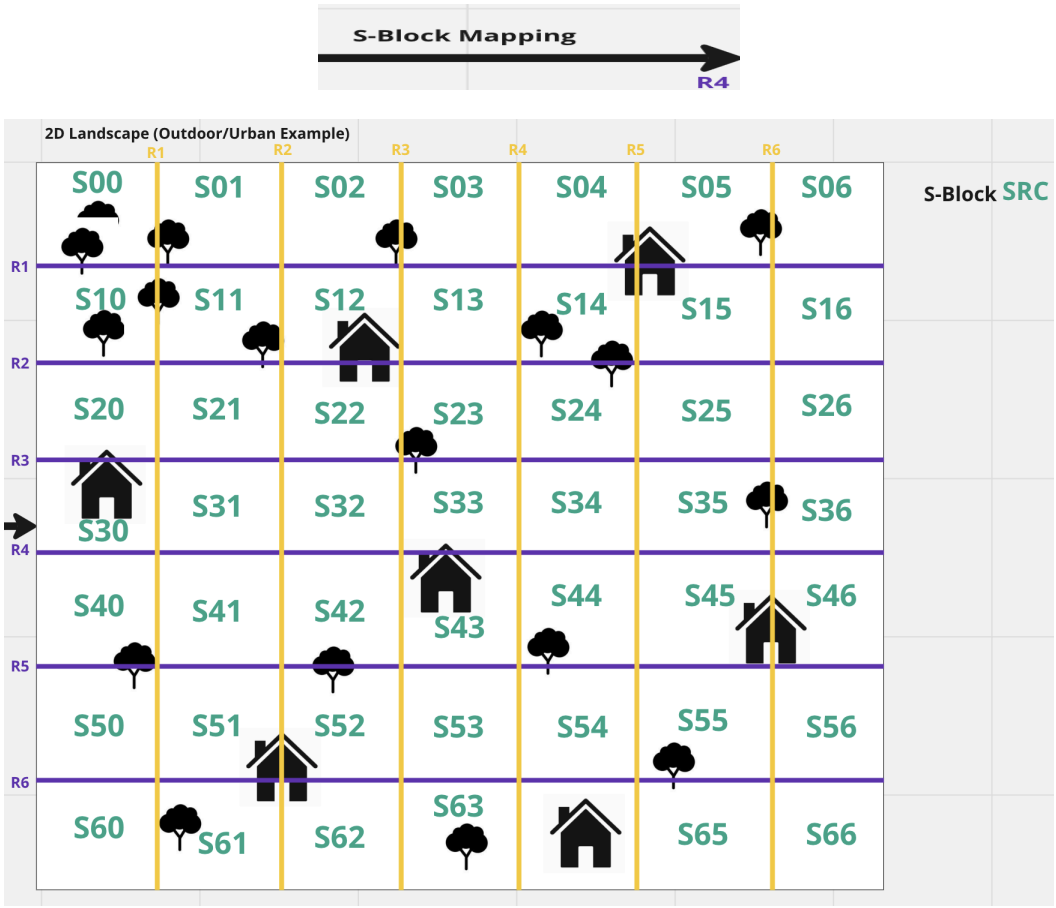

Figure 3: S-Block Mapping

Spatial Compute With SIB Simulation (Idea)

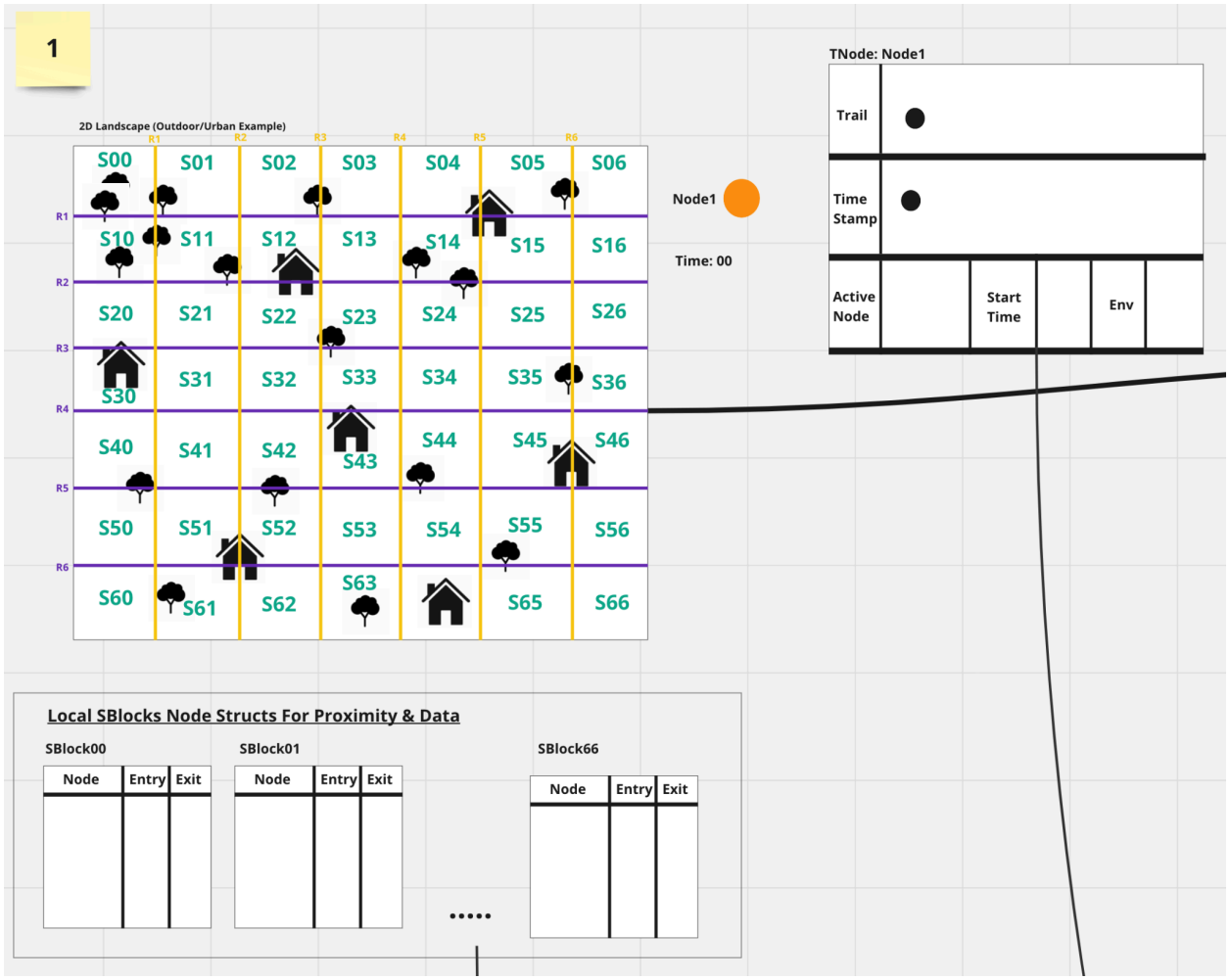

Figure 4: Initial Environment With Node1 To Enter

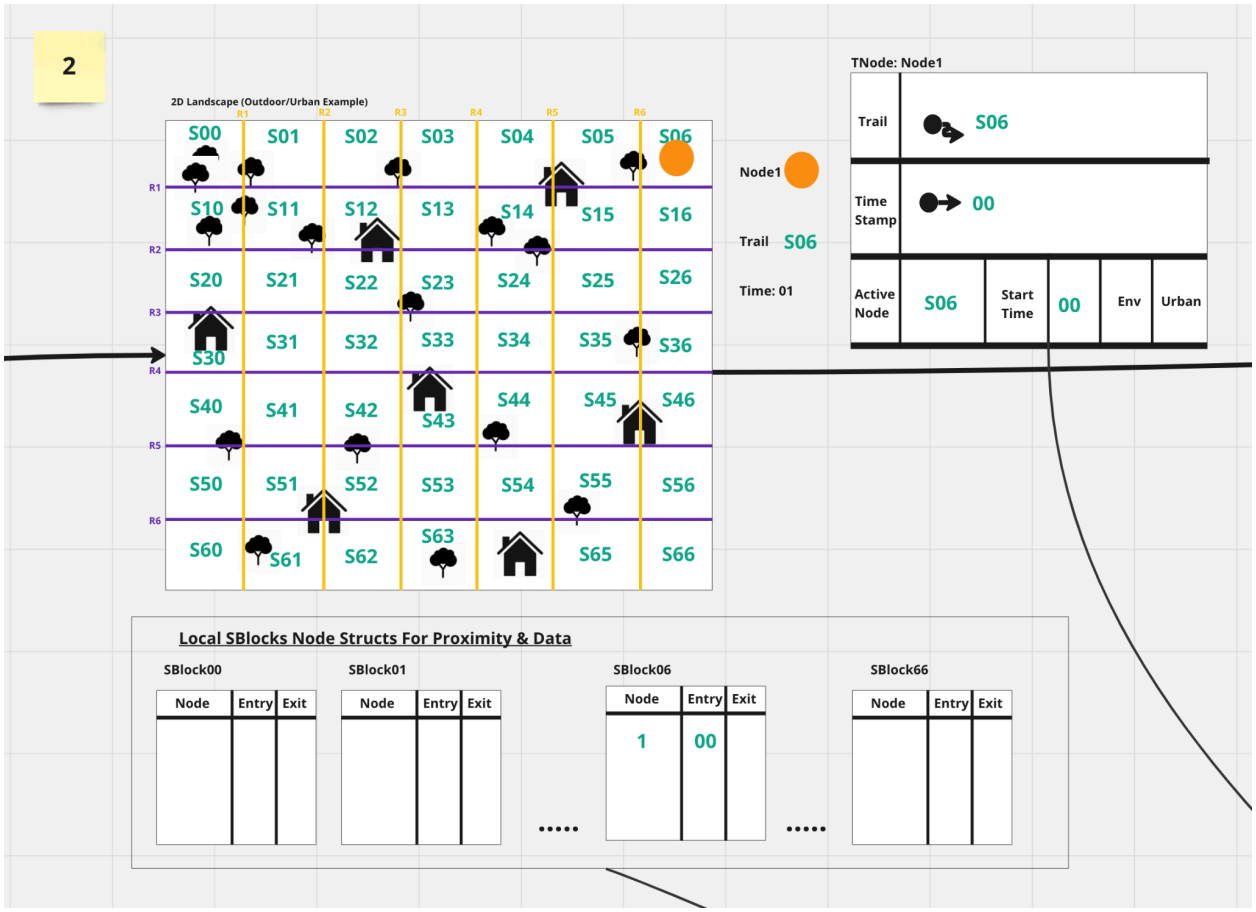

Figure 5: Node1 Engage With Env, Change In SBlock, Trail, TNode Shown

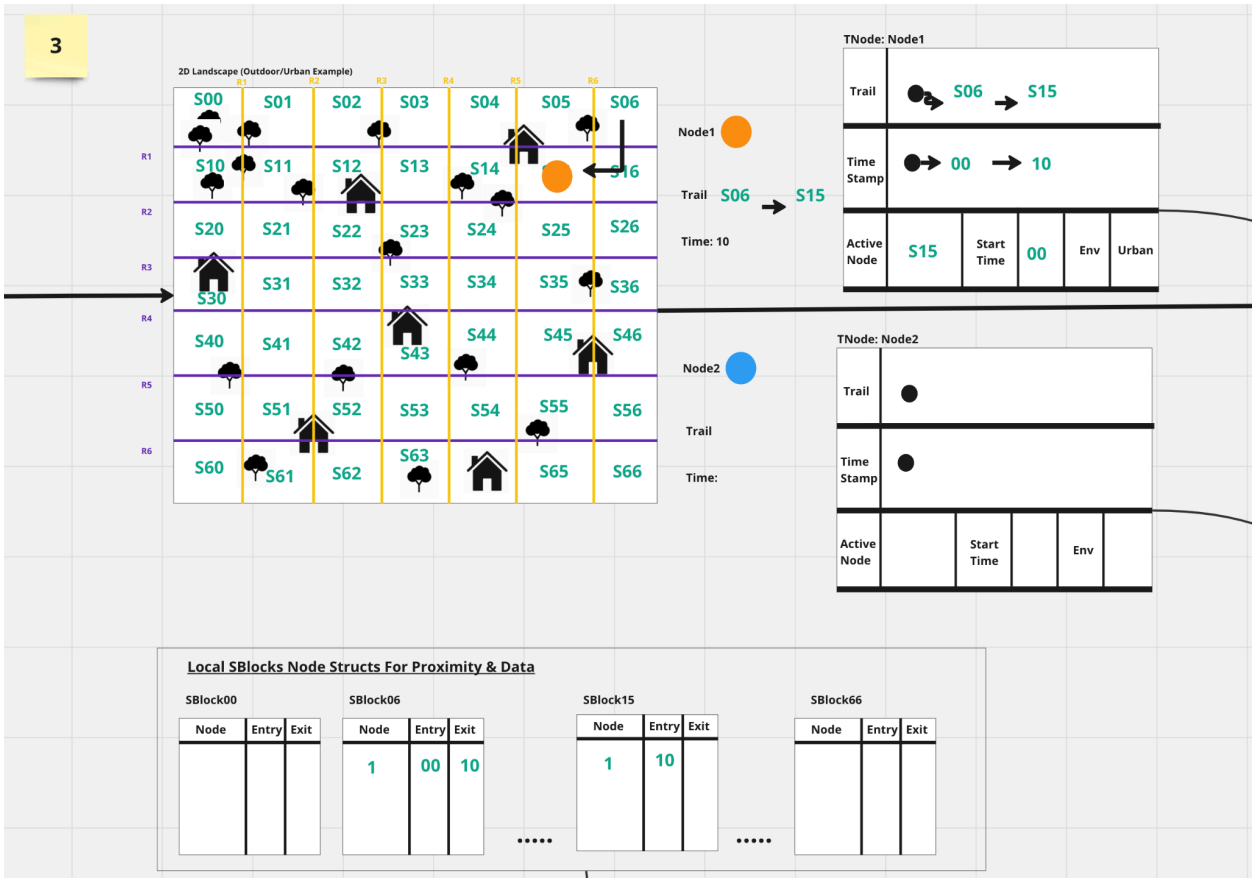

Figure 6: Node1 Engage With Env, Change In SBlock, Trail, TNode Shown For Node1, Node2 Positioned To Enter Env

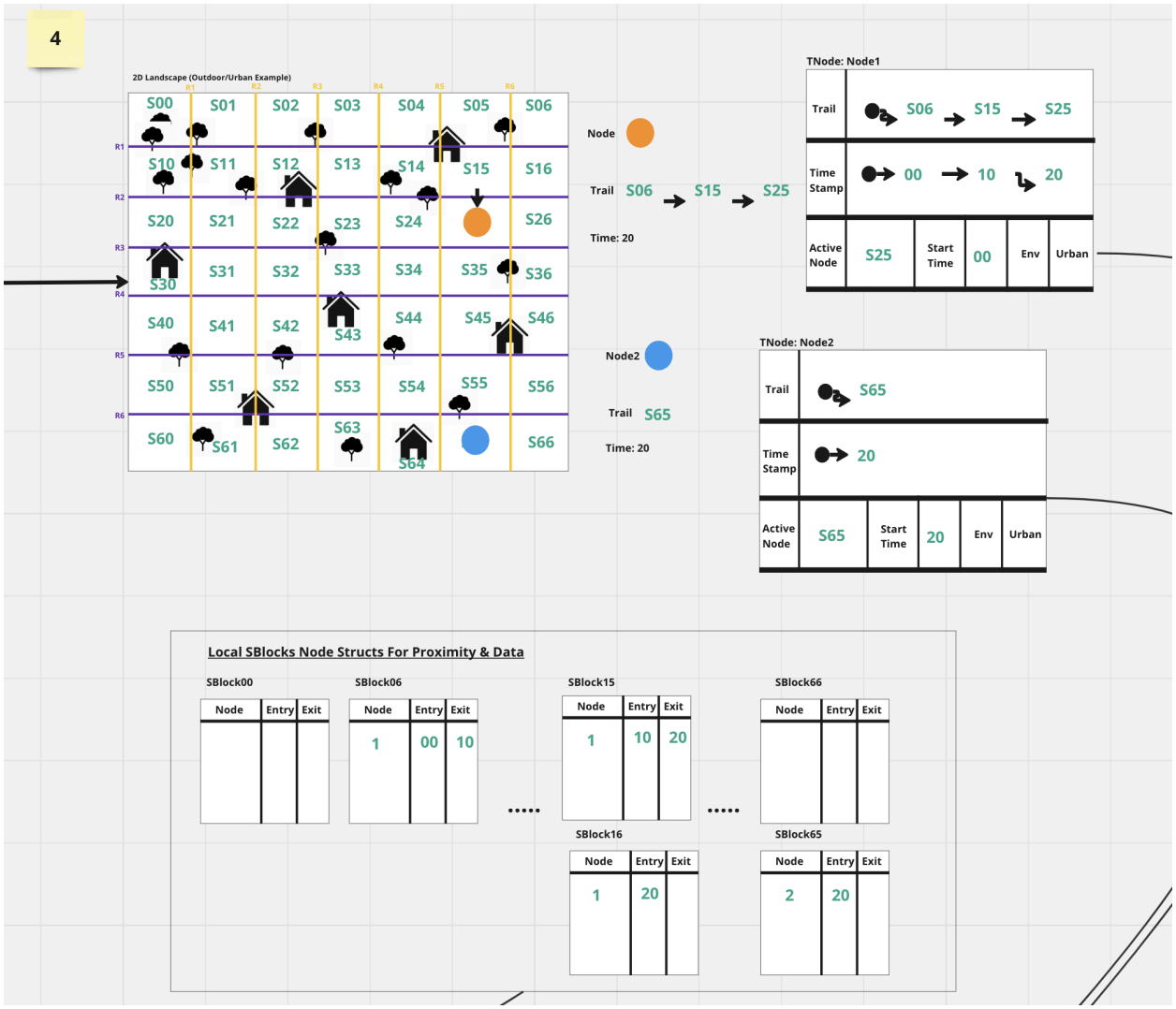

Figure 7: Node1 Engage With Env, Change In SBlock, Trail, TNode Shown For Node1, Node2 Enage With Env, Change In SBlock, Trail, TNode Shown For Node2

Figure 8: Local Data Being Sent To Cloud For Storage + Analytics