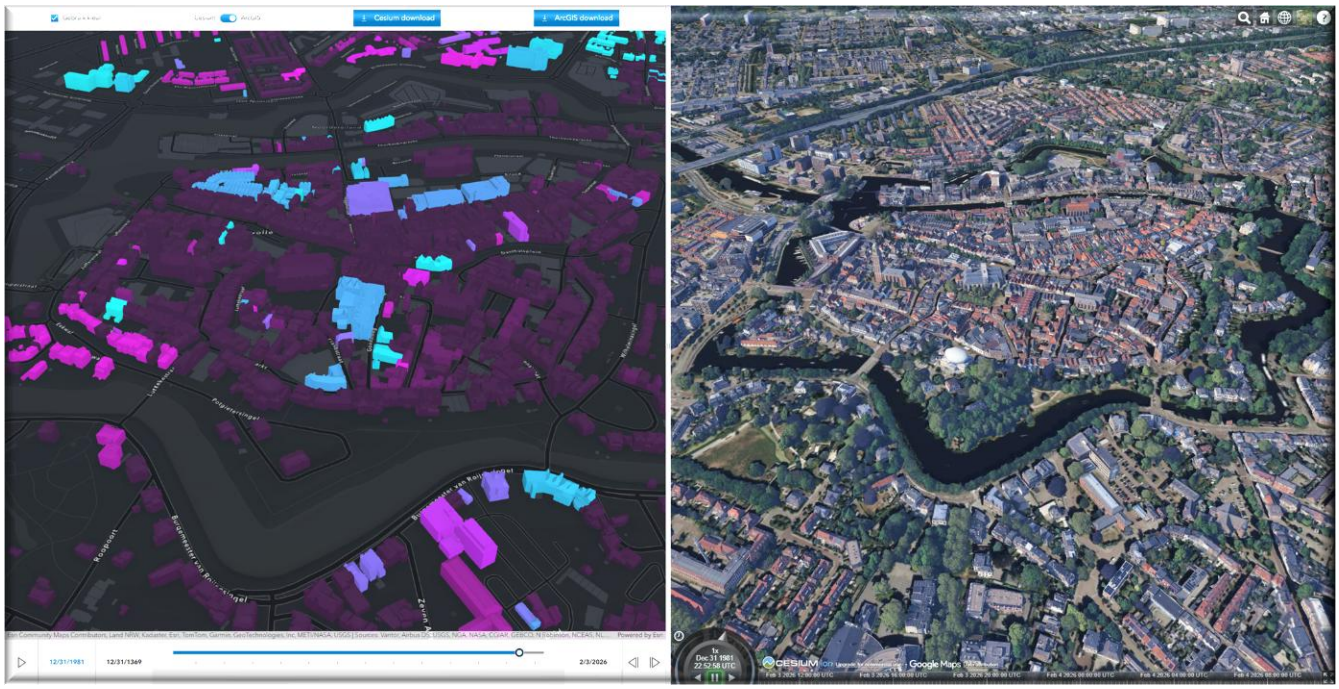
Digital Twin as a Service – Testbed II

Import & Export Scenes



esri Nederland
THE SCIENCE OF WHERE

Inhoudsopgave

Executive Summary

Visualization applications lack a standardized method for exchanging scene configurations, potentially leading to inefficiencies and limited collaboration between Digital Twins. This report addresses the challenge of import and export of 3D scenes within the context of Digital Twin Testbed 2. Through analysis of existing standards, schema design, and practical demonstrations with real-world data, this research proposes a portable scene context schema, validated between ArcGIS and Cesium environments. The findings could lay some exploratory groundwork for a standards-based, future-proof solution, and provide actionable recommendations for advancing interoperability in any Digital Twin ecosystem.

Note on Scope and Value:
The importance of scene exchange lies in fostering collaboration, consistency, and reproducibility within and across digital twin environments. Scene exchange does not mean that every visualization platform or application must replicate the full functionality of another; rather, it is about sharing a common foundation of scene elements. This includes layers, camera viewpoints, temporal settings, styling, and basic interactions. By standardizing the exchange of these elements, interoperability is enabled without diminishing the unique strengths of each application.

A standard for scene exchange offers several key benefits:

- **Interoperability:** Scenes can be shared and reconstructed across different systems.
- **Reproducibility:** Analyses and visualizations can be exactly repeated or audited in the future.
- **Documentation:** Scene configurations can be archived as part of project records or reports.

The goal is not to require complete feature parity between platforms, but to establish a shared baseline that supports exchange and (long-term) collaboration.

1 Introduction

The field of digital twins is developing rapidly, with platforms and applications such as ArcGIS and Cesium providing robust environments for spatial analysis and 3D data exploration. For many organizations and users, working within a single, well-integrated platform offers all the tools and capabilities needed for effective visualization, analysis, and decision-making. ArcGIS, for example, delivers a mature and comprehensive suite of features that support a wide range of workflows and stakeholder needs.

The ambition of the Dutch Digital Twin Testbed 2 is to explore how digital twins can become more connected, flexible, and future proof. In this context, the question of interoperability and scene exchange sometimes arises, especially as digital twin initiatives become more collaborative and multidisciplinary, or when federated systems are envisioned. There are scenarios, such as cross-organizational projects, long-term reproducibility, or the desire to leverage the unique strengths of different applications, where the ability to exchange 3D scene configurations can add value. In these cases, interoperability is not an end in itself, but a means to support collaboration, consistency, and sustainability where it is genuinely required.

It is important to emphasize that scene exchange is not always necessary, nor should it be seen as a requirement for every project or platform. Each application has its own strengths and specializations, and many users will continue to benefit from the deep integration and advanced capabilities of their chosen environment. The aim of this research is not to diminish those strengths, but to explore how a shared baseline for scene exchange can be established, enabling the transfer of essential scene elements such as layers, camera viewpoints, temporal settings, and basic styling, in situations where this supports the broader goals of a project.

This report investigates the possibilities and limitations of scene exchange within the context of Digital Twin Testbed 2. It analyzes existing standards, proposes a portable scene context schema, and validates this approach in practice between ArcGIS and Cesium. The results are intended to provide guidance for situations where interoperability is desirable, while respecting the value and robustness of individual platforms.

2 Background & Standards

Digital twins are increasingly recognized as a key enabler for integrated, data-driven decision-making in the built environment. As the Dutch Digital Twin Testbed 2 demonstrates, the ambition is not only to build powerful tools, but also to ensure that these tools can work together, when and where it adds value. This chapter provides an overview of the standards, specifications, and practical approaches relevant to scene exchange, and illustrates what a portable scene configuration might look like in practice.

2.1 Scene Exchange in Visualization Platforms and Applications

Most visualization platforms and applications, including the ArcGIS platform and Cesium platform, use their own scene configuration formats. A visualization platform provides the broader infrastructure, APIs, and integration capabilities for spatial data and digital twin solutions, while a visualization application, such as ArcGIS Scene Viewer or a CesiumJS web client, enables users to interact with, configure, and visualize 3D scenes. These applications typically capture the essential elements of a 3D visualization: camera position, orientation, time settings, and a stack of layers with their sources and visibility. While this approach works well within a single application or platform, it can create barriers when there is a need to share or reconstruct a scene in another environment, whether for collaboration, reproducibility, or leveraging unique features of different platforms.

Testbed 2 explicitly explores the potential of federated digital twin systems, where interoperability is not a goal in itself, but a means to support multi-disciplinary collaboration and long-term sustainability. In this context, scene exchange is about establishing a shared baseline for what can be transferred between platforms and applications, not about achieving full feature parity.

2.2 Relevant Standards and Specifications

In Digital Twin Testbed 2, the import and export of scenes focuses on exchanging the configuration of layers, camera, time, and data references between visualization applications such as ArcGIS and Cesium. The following standards are relevant:

- **ArcGIS Web Scene Specification**

The ArcGIS Web Scene Specification is a JSON-based format that fully describes a 3D scene within the ArcGIS platform. It includes all essential configuration elements such as:

  - **Layers:** Basemaps, elevation layers, and operational layers (e.g., 3D buildings, thematic data).
  - **Camera and Viewpoint:** Position, heading, tilt, and zoom level.
  - **Environment Settings:** Lighting, atmosphere, and background.
  - **Styling and Popups:** Symbology, color schemes, and interactive elements.

This specification is widely used in the ArcGIS ecosystem and serves as a practical reference for exporting and importing scenes. It ensures that a scene can be consistently reconstructed in any ArcGIS application and provides a clear structure for mapping to other platforms.

- **CesiumJS Scene Model**

CesiumJS uses a JavaScript-based configuration model to initialize and render 3D scenes in a web environment. Key elements include:

  - **Camera Settings:** Position, orientation, and movement controls.
  - **Data Sources:** Support for imagery layers, terrain, and vector data.
  - **3D Content:** Integration of 3D Tiles for large-scale datasets and primitives for custom geometries.
  - **Time and Animation:** Handling of temporal data for dynamic visualizations.

The Cesium scene model is flexible and optimized for high-performance visualization, making it suitable for streaming large datasets and interactive applications. For scene exchange, its structure provides a basis for mapping essential elements like viewpoint and layers to a portable schema.

- **OGC WMS (Web Map Service)**

OGC WMS is a widely adopted standard for serving georeferenced map images over the web. It allows visualization applications to request and display map layers from remote servers using a simple URL-based interface. Key features:

  - **Layer Access:** Clients can request specific layers and styles.
  - **Coordinate Reference Systems:** Support for multiple projections.

WMS layers can be integrated into both ArcGIS and Cesium, making them a common component in scene configurations. While WMS is primarily 2D, it remains relevant for background maps or thematic overlays in 3D scenes, ensuring that shared layers can be consistently displayed across platforms. Scene import and export in Testbed 2 is about transferring the essential configuration of a visualization, supported by open standards. This ensures that scenes can be shared, reconstructed, and reused across different applications and organizations within the testbed.

2.3 What Does a Portable Scene Look Like?

A portable scene configuration captures the essential elements needed to reconstruct a visualization in another platform. Below are two real-world inspired examples, illustrating the kind of data that can be exchanged between ArcGIS and Cesium:

Example 1: Minimal Scene

```
{
  "view": {
    "heading": 90.0,
    "tilt": 45.0,
    "position": {
      "lat": 51.5,
      "lon": 5.0,
      "z": 300.0
    }
  },
  "time": {
    "current": "2023-06-01T09:00:00Z",
    "start": "2023-06-01T09:00:00Z",
    "end": "2023-06-01T09:00:00Z"
  },
  "layers": [
    {
      "title": "Imagery Layer",
      "index": 0,
      "visible": true,
      "type": "imagery",
      "url": "https://example.com/imagery/service"
    }
  ]
}
```

Example 2: Richer Scene with Multiple Layers

```
{
  "view": {
    "heading": 45.0,
    "tilt": 60.0,
    "position": {
      "lat": 52.0,
      "lon": 6.0,
      "z": 500.0
    }
  },
  "time": {
    "current": "2025-01-01T12:00:00Z",
    "start": "2020-01-01T00:00:00Z"
  },
  "layers": [
    {
      "title": "Base Map",
      "index": 0,
      "visible": true,
      "type": "vector-tile",
      "url": "https://example.com/basemap/style.json"
    },
    {
      "title": "Elevation",
      "index": 1,
      "visible": true,
      "type": "elevation",
      "url": "https://example.com/elevation/service"
    },
    {
      "title": "3D Buildings",
      "index": 2,
      "visible": true,
      "type": "scene",
      "url": "https://example.com/3d/buildings"
    }
  ]
}
```

These examples show how a scene can be described in a platform-agnostic way, capturing the viewpoint, time, and a stack of layers with their types and sources. Such a schema can be mapped to both ArcGIS and Cesium, enabling practical scene exchange where it is needed.

2.4 Summary

The standards landscape provides a solid foundation, but not a complete solution, for scene exchange. By building on these standards and focusing on a practical, schema-based approach, Testbed 2 aims to demonstrate how scene configurations can be made portable—supporting interoperability where it adds value, while respecting the strengths of individual platforms.

3 Research Questions

This research addresses the practical challenge of exchanging 3D scene configurations between visualization applications in Digital Twin Testbed 2. The goal is to define a minimal, portable approach that supports interoperability without requiring full feature parity between platforms.

The guiding questions are:

3.1 What is the minimal, client-agnostic schema for scene exchange?

Which elements are essential to describe a scene, such as layers, camera viewpoint, time settings, and basic styling, so that it can be interpreted by different applications?

3.2 How can resources be referenced and mapped across platforms?

What approach enables external resources (e.g., data services, styles) to be referenced in a portable way, and how can differences between ArcGIS and Cesium formats be reconciled?

3.3 What are the practical and technical barriers?

Which limitations arise from differences in functionality, interoperability, and resource accessibility, and how do these affect the feasibility of a shared baseline?

The objective of this report is to explore how 3D scene configurations can be exchanged in a way that is practical, lightweight, and future-proof. These questions directly support that goal by:

- Defining a minimal schema for interoperability without unnecessary complexity.
- Investigating how resources can be referenced and mapped for portability.
- Identifying barriers that influence feasibility and next steps.

Together, these questions form the foundation for proposing a portable scene context approach that aligns with the ambitions of Digital Twin Testbed 2: enabling collaboration, reproducibility, and flexibility in a federated digital twin ecosystem.

4 Methodology

The research approach is practical and exploratory, aimed at validating whether a portable scene context can work in the context of Digital Twin Testbed 2. The process consisted of three main steps:

4.1 Review of Existing Formats and Standards

The first step was to analyze how scenes are currently described in visualization applications and what standards exist that could support exchange. This included:

- ArcGIS Web Scene Specification and CesiumJS Scene Model for platform-specific configurations.
- OGC OWS Context as a conceptual basis for packaging resources.
- Metadata and API standards such as OGC API – Records and DCAT-AP-NL for resource referencing.

The purpose of this review was to identify common elements and determine what is essential for a minimal, client-agnostic schema.

4.2 Design of a Portable Scene Schema

Based on the analysis, a lightweight JSON schema was drafted to capture the core components of a scene:

- Viewpoint: Camera position, heading, tilt.
- Time: Current time and optional temporal range.
- Layers: Title, type, visibility, and URL for each layer.

The schema was designed to be platform-neutral but still allow mapping to ArcGIS and Cesium formats.

4.3 Proof-of-Concept Demonstration

The schema was validated through a practical test:

- Exporting a scene from ArcGIS and converting it into the portable schema.
- Importing the schema into a Cesium-based application to reconstruct the scene.
- Testing with real-world data and multiple layer types (basemap, elevation, 3D objects).

The demonstration was implemented using web technologies (Vite, Azure Static Web Apps) to ensure reproducibility and easy sharing.

5 Schema Design

A key outcome of this research is the proposal for a portable scene context schema (SCX). The schema is designed to capture the essential elements of a 3D scene in a platform-neutral way, enabling import and export between visualization applications without requiring full feature parity.

5.1 What is SCX

Current scene formats, such as ArcGIS Web Scene and CesiumJS Scene Model, are platform-specific and include advanced features that are not universally supported. SCX focuses on a minimal set of elements that are common across platforms:

- Viewpoint: Camera position and orientation.
- Time: Current time and optional temporal range.
- Layers: Basic properties (title, type, visibility, URL).

This approach ensures interoperability while keeping the schema lightweight and easy to implement. In a broader context, efforts within the OGC community, such as OWS Context, which offers a structured way to package and exchange geospatial resource configurations across applications, illustrate ongoing work toward portable, standards-based context representations.

5.2 Schema Structure

The SCX schema is organized into three main sections:

Section	Description
View	Camera settings: position (latitude, longitude, altitude), heading, tilt
time	Current time and optional start/end range
layers	Array of layer objects with title, type, visibility, and URL

5.3 Generic JSON Example

```
{
  "view": {
    "heading": 45.0,
    "tilt": 60.0,
    "position": {
      "latitude": 52.0,
      "longitude": 6.0,
      "altitude": 500.0
    }
  },
  "time": {
    "current": "2025-01-01T12:00:00Z",
    "start": "2020-01-01T00:00:00Z",
    "end": "2030-01-01T00:00:00Z"
  },
  "layers": [
    {
      "title": "Base Map",
      "type": "vector-tile",
      "visible": true,
```

```
      "url": "https://example.com/basemap/style.json"
    },
    {
      "title": "Elevation",
      "type": "elevation",
      "visible": true,
      "url": "https://example.com/elevation/service"
    },
    {
      "title": "3D Buildings",
      "type": "scene",
      "visible": true,
      "url": "https://example.com/3d/buildings"
    }
  ]
}
```

Note: in the examples coordinates follow WGS84 (EPSG:4326).

5.4 Mapping Table: SCX ↔ ArcGIS ↔ Cesium

SCX Element	ArcGIS Web Scene	CesiumJS Scene Model
view.heading	camera.heading	viewer.scene.camera.heading
view.tilt	camera.tilt	camera.pitch
position.latitude	camera.position.y	Cartographic.latitude
position.longitude	camera.position.x	Cartographic.longitude
position.altitude	camera.position.z	Cartographic.height
time.current	timeSlider.currentTime	Clock.currentTime
layers.title	layer.title	ImageryLayer does not have a native name property (must be managed in app code)
layers.url	layer.url	imageryProvider.url

5.5 Why This Matters

This schema provides a practical foundation for scene exchange. It is simple enough to implement yet flexible enough to support common visualization needs across platforms. By focusing on essential elements, SCX avoids unnecessary complexity while enabling interoperability where it adds value.

6 Demonstration & Results

The proof-of-concept demonstrates that a portable scene context schema (SCX) is a practical starting point for enabling scene exchange between visualization applications. It confirms that interoperability is achievable when focused on essential elements, but several considerations remain.

6.1 Interoperability vs. Feature Parity

SCX successfully captured core components; viewpoint, time, and layers, while intentionally excluding advanced features such as pop-ups, complex symbology, and environmental effects. Full feature parity between platforms is unrealistic seems unnecessary for this experiment. Instead, SCX provides a shared baseline that supports collaboration and reproducibility without compromising platform-specific strengths.

6.2 Mapping Complexity

Mapping between ArcGIS Web Scene and CesiumJS formats worked for basic elements but revealed differences:

- Units: ArcGIS uses degrees for heading and tilt; Cesium uses radians.
- Styling: ArcGIS supports detailed symbology; Cesium requires custom logic for similar effects.
- Layer metadata: Cesium lacks native support for layer titles, requiring application-level handling.

These differences highlight the need for clear documentation and possibly helper libraries to simplify conversion.

6.3 Technical Barriers

Several challenges remain:

- Authentication: Scenes referencing secured services require token handling.
- Performance: Large datasets may need tiling or streaming for efficient rendering.

6.4 Implications for Testbed Goals

The findings support the testbed's objectives:

- Collaboration: SCX enables sharing of scene configurations across platforms.
- Reproducibility: Capturing essential elements in a portable format allows scenes to be reconstructed later.
- Scalability: The schema can evolve with optional extensions for advanced features without breaking compatibility.

6.5 Demonstration

The demo was implemented as a web-based proof-of-concept, running in a lightweight development environment:

- Platform: A simple web application built with Vite as the bundler and CesiumJS for visualization.
- Hosting: Deployed on Azure Static Web Apps for easy access during the live demonstration.
- Workflow:
    - ArcGIS Web Scene JSON was retrieved via the ArcGIS REST API.
    - Conversion scripts (ArcGIS → SCX → Cesium and vice versa) were executed client-side in JavaScript.
    - Cesium viewer rendered the scene using SCX data.
    - For the reverse flow, Cesium camera and layer data were extracted and mapped back to an ArcGIS Web Scene structure.

- A live demonstration of the SCX conversion and rendering workflow was presented during the testbed session, showing how SCX can function as a lightweight bridge between visualization environments without requiring server-side processing. The demonstration is accessible via the following link: Arcesium

  Because the long-term availability of this online demo cannot be guaranteed, supplementary materials, such as shareable video recordings, are provided to ensure continued access to the demonstration. For organizations or individuals who require additional insight, this can be arranged upon request.

6.6 Summary

The demonstration shows that scene exchange is achievable when limited to a concise set of essential scene elements. Rather than attempting to replicate all platform-specific functionality, the SCX approach focuses on preserving the core intent of a scene in a lightweight and interoperable manner. This reinforces the value of a minimal schema for cross-platform portability.

The conversion workflow used in the demo, covering the exchange path between ArcGIS, SCX and Cesium, is documented in Appendix A: How to Reproduce the Scene Exchange Demo, providing step-by-step guidance for reproducing the process.

7 Discussion

The proof-of-concept shows that a portable scene context schema (SCX) can enable basic scene exchange between visualization applications. It demonstrates that interoperability is possible without requiring full feature parity, which aligns with the goals of Digital Twin Testbed 2.

7.1 Effectiveness of SCX

SCX successfully captured essential elements, viewpoint, time, and layers; and allowed reconstruction of a scene in both ArcGIS and Cesium environments. This validates the idea of a shared baseline for scene exchange, supporting collaboration and reproducibility.

7.2 What Works and What Doesn't

Interoperable elements:

- Camera viewpoint (heading, tilt, position)
- Basic time settings (current time, optional range)
- Layer references (title, type, URL)
- Visibility flags for layers

Not easily interoperable and remaining gaps:

- Advanced symbology and styling
- Pop-ups and interactive elements
- Environment settings (lighting, atmosphere)
- Platform-specific features like ArcGIS presentation slides or Cesium primitives

These limitations suggest SCX should remain focused on essentials, with optional extensions for advanced features.

7.3 Broader Context

Although SCX is a small technical step, it contributes to the larger ambition of interoperability in digital twins. It moves from platform-specific workflows toward a shared baseline and lays the foundation for more advanced orchestration in the future. However, this report only explores the first possibilities.

8 Recommendations & Next Steps

This work is an initial exploration of scene exchange in Digital Twin Testbed 2. It shows that a minimal schema (SCX) can support basic interoperability, but it is far from complete. Many aspects still need research and refinement.

First, the concept of SCX could be shared openly, with clear definitions and examples. It must remain minimal but allow optional extensions for advanced features. Publishing a reference implementation will help gather feedback and encourage collaboration.

Second, simple conversion tools are needed to platform specific configurations to SCX and back. Sharing these tools as-is will make it easier for others to experiment and contribute.

Third, SCX should be aligned with existing and emerging standards. Working with OGC and Geonovum will clarify if and how a concept like SCX could evolve into a profile or standard.

Finally, SCX should be tested in more workflows and refined step by step.

8.1 Conclusion

SCX is not a finished solution, it is a starting point. To move toward robust interoperability in the Dutch Digital Twin ecosystem, scene resources must be discoverable and accessible.

This research demonstrates that a minimal, client-agnostic schema for scene exchange is feasible for basic interoperability. Many gaps remain, including styling, advanced layer support, authentication, and performance. Further research and collaboration are essential. If these steps are taken, the Dutch Digital Twin ecosystem can evolve toward robust, standards-based interoperability.

Appendix: How to Reproduce the Scene Exchange Demo

This appendix documents the core logic used in the proof-of-concept to exchange scene configurations between ArcGIS, SCX, and Cesium. These scripts demonstrate how essential elements (camera, time, layers) were mapped in both directions.

The conversion scripts provided in this appendix are simplified examples intended to illustrate the core logic of the interoperability demo. They do not represent the full complexity of the live implementation.

1. ArcGIS → SCX
Extract key properties from an ArcGIS Web Scene JSON and convert them into SCX format:

```
function arcgisToSCX(arcgisScene) {
  const camera = arcgisScene.camera;
  const layers = arcgisScene.operationalLayers || [];
  return {
    view: {
      heading: camera.heading,
      tilt: camera.tilt,
      position: {
        latitude: camera.position.y,
        longitude: camera.position.x,
        altitude: camera.position.z
      }
    },
    time: {
      current: arcgisScene.timeSlider?.currentTime || null
    },
    layers: layers.map(layer => ({
      title: layer.title,
      type: layer.layerType || "unknown",
      visible: layer.visibility,
      url: layer.url
    }))
  };
}
```

2. SCX → Cesium
Apply SCX values to a Cesium viewer:

```
function applySCXToCesium(viewer, scx) {
  viewer.camera.setView({
    destination: Cesium.Cartesian3.fromDegrees(
      scx.view.position.longitude,
      scx.view.position.latitude,
      scx.view.position.altitude
    ),
    orientation: {
      heading: Cesium.Math.toRadians(scx.view.heading),
      pitch: Cesium.Math.toRadians(-scx.view.tilt)
    }
  });
  scx.layers.forEach(layer => {
    viewer.imageryLayers.addImageryProvider(new Cesium.UrlTemplateImageryProvider({
      url: layer.url
    }));
```

```
  });
}
```

3. Cesium → SCX
Extract Cesium camera and layer info into SCX:

```
function cesiumToSCX(viewer) {
  const carto = Cesium.Cartographic.fromCartesian(viewer.camera.position);
  return {
    view: {
      heading: Cesium.Math.toDegrees(viewer.camera.heading),
      tilt: -Cesium.Math.toDegrees(viewer.camera.pitch),
      position: {
        latitude: Cesium.Math.toDegrees(carto.latitude),
        longitude: Cesium.Math.toDegrees(carto.longitude),
        altitude: carto.height
      }
    },
    layers: viewer.imageryLayers._layers.map(layer => ({
      title: "Cesium Layer",
      type: "imagery",
      visible: true,
      url: layer.imageryProvider.url
    }))
  };
}
```

4. SCX → ArcGIS
Convert SCX back to ArcGIS Web Scene JSON:

```
function scxToArcGIS(scx) {
  return {
    camera: {
      position: {
        x: scx.view.position.longitude,
        y: scx.view.position.latitude,
        z: scx.view.position.altitude,
        spatialReference: { wkid: 4326 }
      },
      heading: scx.view.heading,
      tilt: scx.view.tilt
    },
    operationalLayers: scx.layers.map(layer => ({
      title: layer.title,
      url: layer.url,
      visibility: layer.visible
    }))
  };
}
```

Notes
Layer Metadata: Cesium does not store titles natively; these were added manually.