Das iCity-Framework zur Entwicklung urbaner digitaler Zwillinge

The iCity Framework for the Development of Urban Digital Twins

Volker Coors | Matthias Betz | Thunyathep Santhanavanich

Zusammenfassung
Digitale Zwillinge gewinnen als Teil des kommunalen Geodatenmanagements und der Geodateninfrastruktur von Städten und Kommunen zunehmend an Bedeutung. In diesem Beitrag wird ein Framework zur Entwicklung von urbanen digitalen Zwillingen vorgestellt, das im Rahmen der vom Bundesministerium für Bildung und Forschung geförderten Projektpartnerschaft iCity entwickelt wurde. Die Nutzung des Frameworks wird an drei Fallstudien evaluiert, wobei die Schwerpunkte bei der Integration von Sensordaten und der Nutzung eines 3D-Stadtmodells als Datenbasis für urbane Simulationen liegen.

Schlüsselwörter: Geodateninfrastruktur, urbane digitale Zwillinge, 3D-Stadtmodelle, urbane Simulation

Summary
Urban digital twins are becoming increasingly important as part of municipal geodata management and the geodata infrastructure of cities and municipalities. This publication presents a framework for the development of urban digital twins, which was developed as part of the iCity project partnership funded by the Federal Ministry of Education and Research. The use of the framework is evaluated in three case studies, focussing on the integration of sensor data and the use of a 3D city model as a database for urban simulations.

Keywords: *geodata infrastructure, urban digital twins, 3D city models, urban simulation*

1 Einleitung

Die Entwicklung der Digitalisierung im Zusammenhang mit der Geodateninfrastruktur Deutschland (GDI-DE) wurde in (Ostrau 2024) ausführlich dargestellt. Dabei werden digitale Zwillinge als »Treiber der Entwicklung mit neuer Rolle des [kommunalen] Geodatenmanagements« hervorgehoben. Zahlreiche Städte und Kommunen bauen derzeit digitale Zwillinge im Zuge einer Smart-City-Strategie auf. Mit der DIN SPEC 91607 »Digitaler Zwilling für Städte und Kommunen« (Schubbe et al. 2023) wurde eine Normungsinitiative gestartet, die das Konzept eines digitalen Zwillings, welches seinen Ursprung in der Raumfahrt und dem Maschinenbau hat, auf das komplexe und dynamische Umfeld in Städten und Kommunen übertragen. Diese urbanen digitalen Zwillinge (UDZ) bilden ein digitales Abbild der kommunalen Realität und beinhalten neben einem Modell der bebauten Umgebung auch Prozesse, Prognosemodelle und Planungsszenarien. Aufgrund dieser Komplexität wird es nicht einen allumfassenden UDZ einer Kommune geben, sondern je nach Fragestellung verschiedene fachspezifische Zwillinge. Die Landeshauptstadt Stuttgart entwickelt beispielsweise einen digitalen Zwilling für Mobilität und Umwelt (LHS Stuttgart 2024). Mit dem Kalasatama Digital Twin hat die Stadt Helsinki einen UDZ für die Quartiersentwicklung erstellt (Kalasatama 2019).

Es bedarf also einer Art Baukastensystem, um einen fachspezifischen UDZ möglichst kostengünstig und effizient aus bewährten Komponenten zu erstellen. Diese Komponenten sollten über standardisierte Schnittstellen verfügen, sodass sie austauschbar und einfach kombinierbar sind. Donaubauer et al. (2023) haben im Rahmen der TwinBy-Initiative einen Leitfaden für die Entwicklung von UDZ veröffentlicht. Im Projekt Connected Urban Twins (CUT) wird ein modulares System für die Erstellung von UDZ zur Visualisierung, Analyse und Simulation entwickelt. Als Anwendungsfälle stehen ein Geobasiszwilling, digitale Partizipation, Energienutzungsplanung, KiTa-Netzplanung und die bauplanungsrechtlich geplante Stadt im Vordergrund (Schubbe et al. 2023). Auf europäischer Ebene wurde im Projekt DUET ein Framework für UDZ konzipiert und anhand von Fallstudien im Bereich Verkehrsplanung, Lärmausbreitung und Luftqualität evaluiert (Raes et al. 2022).

Während im DUET-Projekt ein 3D-Stadtmodell nur zur Visualisierung genutzt wird und die integrierte Simulation auf 2D-Geodaten basiert, soll in diesem Beitrag ein Framework für die Entwicklung von fachspezifischen UDZ vorgestellt werden, bei dem 3D-Stadtmodelle auch die Basis für Simulation und Datenanalyse bilden.

2 Urbane digitale Zwillinge

In Anlehnung an die allgemeine Definition eines digitalen Zwillings der Gesellschaft für Informatik (GI 2024) werden in diesem Beitrag urbane digitale Zwillinge als digitale

DOI 10.12902/zfv-0479-2024

Repräsentation von Dingen aus der realen Welt mit Fokus auf die bebaute Umgebung in Städten und Kommunen verstanden. Sie beschreiben sowohl physische Objekte als auch nicht physische Dinge wie Dienste und Prozesse, indem sie alle relevanten Informationen und Dienste mittels einer einheitlichen Schnittstelle zur Verfügung stellen. Für den digitalen Zwilling ist es dabei unerheblich, ob das Gegenstück in der realen Welt schon existiert oder erst existieren wird. Ein übergeordnetes Ziel des UDZ ist die Unterstützung bei der Entwicklung von nachhaltigen Städten und Gemeinden gemäß der Sustainable Development Goals der Vereinten Nationen (SGD 11).

Grundlage eines UDZ ist die Repräsentation der vorhandenen bebauten Umgebung und städtischen Infrastruktur – insbesondere Topografie, Gebäudebestand, Bauwerke, Straßen und Wege, Stadtmöbel, Grünflächen und Gewässer – durch ein raumbezogenes Modell. Dieser Geobasis-Zwilling kann je nach Ausprägung und Abstraktionsgrad die Realität als zweidimensionale Karte oder als dreidimensionales Modell abbilden. In diesem Beitrag wird von einem 3D-Geobasis-Zwilling ausgegangen.

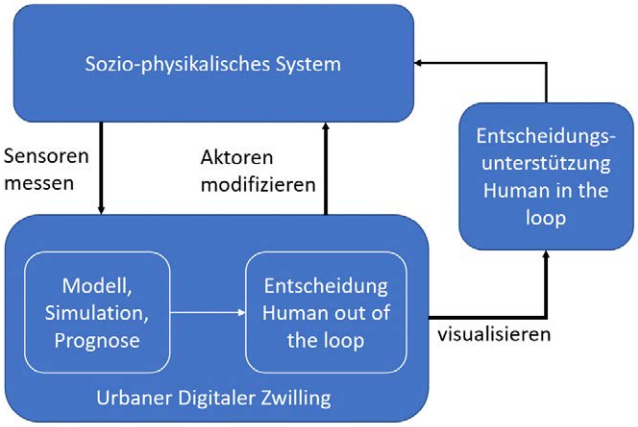


Abb. 1: UDZ als digitales Abbild und Modell eines sozio-physikalischen Systems, der zur datenbasierten Entscheidungsunterstützung dient.

Durch die Integration von Echtzeitdaten aus Sensoren, IoT-Geräten (Internet der Dinge) und anderen Quellen kann ein urbaner digitaler Zwilling zum Monitoring dynamischer Prozesse genutzt werden. Durch Aktoren kann ein UDZ auch Änderungen der realen Welt vornehmen. Ein bekanntes Beispiel aus dem Mobilitätssektor ist das Verkehrsmonitoring und ein adaptives Verkehrsleitsystem, das in Abhängigkeit des Verkehrsaufkommens den Verkehrsfluss steuert, ohne dass ein Mensch eingreift (»Human out of the Loop«). Die meisten UDZ werden aber nicht zu einer voll automatisierten Steuerung führen, sondern Daten analysieren und durch eine geeignete situative interaktive Visualisierung als Entscheidungsunterstützung dienen. Dieses Prinzip wird als »Human in the Loop« bezeichnet und ist in Abb. 1 dargestellt.

Ein wesentliches Element eines UDZ ist die integrierte Datenanalyse. Bei der Datenanalyse werden i. d. R. deskriptive, diagnostische, prädiktive und präskriptive Verfahren unterschieden. Die deskriptive Datenanalyse dient der Beschreibung erhobener Daten beispielsweise durch Kennzahlen und visuelle Darstellung und versucht, Muster und Trends in den Daten zu erkennen. Darauf aufbauend hat die diagnostische Analyse zum Ziel, Gründe für bestimmte Trends in den Daten zu erkennen. Klassische Verfahren zu deskriptiver und diagnostischer Analyse raumbezogener Daten sind in Geoinformationssystemen integriert und etabliert. Insbesondere zur diagnostischen Analyse kommen mehr und mehr Verfahren mit Unterstützung künstlicher Intelligenz (KI) zum Einsatz.

Die prädiktive Analyse beschäftigt sich damit, aus Modellen und historischen Daten zukünftige Ereignisse hervorzusagen. Hierzu werden Simulationen von »What if«-Szenarien und Deep-Learning-Verfahren aus der KI genutzt. Darüber hinaus wird durch präskriptive Analyse versucht, aus den möglichen zukünftigen Ereignissen, die i. d. R. mittels Prädiktion ermittelt wurden, das Optimum bzw. die besten Optionen zu identifizieren und Entscheidungsunterstützung zu liefern, um dieses Optimum zu erreichen.

Für den UDZ bedeutet dies insbesondere, dass die Daten für Analysezwecke aufbereitet werden müssen und eine enge Integration von Datenmanagement und Datenanalyse vorhanden ist. Um zu beschreiben, dass Daten mit geringem Aufwand für Analysen verwendet werden können, wurde in der satellitengestützten Erdbeobachtung der Begriff »Analysis Ready Data« (ARD) geprägt.

Ein komplexes System wie eine Stadt wird nicht durch einen einzigen UDZ abgebildet werden. Das Gesamtsystem besteht aus verschiedenen Sub-Systemen bzw. fachspezifischen Systemen. Um die Komplexität zu reduzieren, ist es sinnvoll, fachspezifische Systeme jeweils in einem fachspezifischen UDZ abzubilden. Dabei ist entscheidend, die verschiedenen fachspezifischen UDZ auf einer gewissen Abstraktionsebene über standardisierte Schnittstellen miteinander zu vernetzen. Dabei soll nicht jedes Detail eines fachspezifischen UDZ geteilt werden, sondern nur aggregierte Informationen, die für andere fachspezifische UDZ relevant sind. Dies hat den Vorteil, dass fachspezifische Datenmodelle nicht in allen Details harmonisiert werden müssen, sondern nur ein Kerndatenmodell. Dieses Kerndatenmodell wird City Information Modell (CIM) genannt. Die Anpassung der fachspezifischen Semantik muss natürlich nach wie vor erfolgen.

In einem föderierten UDZ können fachspezifische UDZ über standardisierte Schnittstellen anwendungsspezifisch zusammengeführt werden. »Föderiert« bedeutet hierbei, dass die Daten nicht kopiert werden, um eine redundante Datenhaltung zu vermeiden. Auch fachspezifische UDZ sind bereits föderierte UDZ, da sie auf einen gemeinsamen Geobasis-Zwilling zugreifen, der die bebaute Umgebung repräsentiert und durch fachspezifische Elemente angereichert wird. Dies ist in Abb. 2 skizziert. Um föderierte UDZ zu implementieren, bedarf es persistenter Identifikatoren aller relevanten Objekte, die sich auch durch

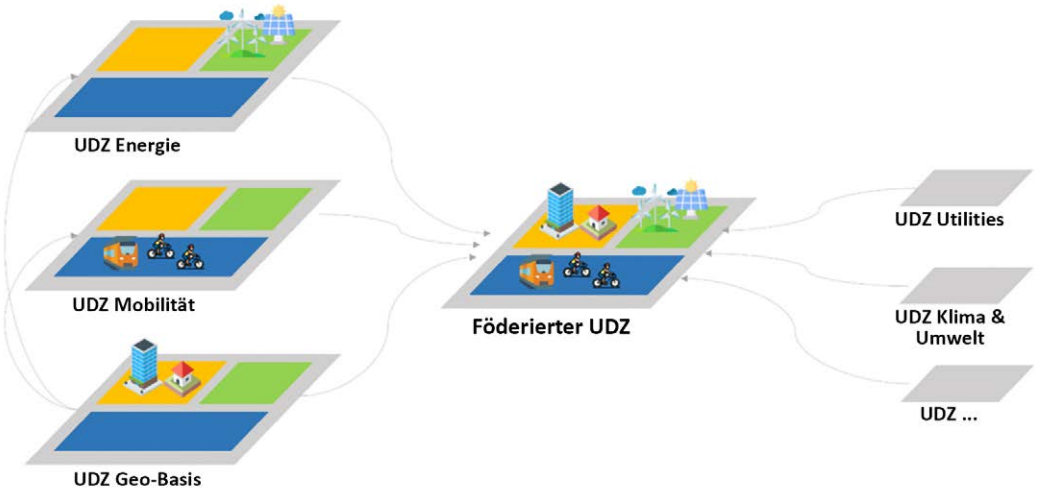


Abb. 2: Föderierte urbane digitale Zwillinge

Aktualisierungen des Datenbestands nicht ändern. Diese persistenten Identifier sind notwendig, um Daten zu verlinken. Dies ist in Deutschland durch den Bezug zum Kataster für Gebäude sichergestellt, allerdings beispielsweise nicht für Dachflächen eines Gebäudes.

3 Konzept des iCity-Frameworks

Im Folgenden wird das Konzept des iCity-Frameworks für die Entwicklung UDZ vorgestellt. Ziel des Frameworks ist es, eine Softwarearchitektur aus modularen Komponenten mit standardisierten Schnittstellen zu entwickeln, um möglichst effizient fachspezifische UDZ zu implementieren. Als Grundlage dienen dabei die Standards des Open-Geospatial-Consortiums (OGC), insbesondere die Entwicklungen der OGC APIs.

In der Software-Entwicklung bezeichnet ein Rahmenwerk bzw. Framework »eine Menge kooperierender Klassen, die unter Vorgabe eines Ablaufes eine generische Lösung für eine Reihe ähnlicher Aufgabenstellungen bereitstellen« (Oestereich 2004) und somit eine Softwarearchitektur vorgeben. Aus Gründen der Übersichtlichkeit werden in dieser Publikation nur die Komponenten dargestellt, die zur Modularisierung genutzt werden. Auf eine detaillierte Beschreibung der einzelnen Komponenten wird an dieser Stelle verzichtet. Abb. 3 skizziert und gruppiert die wesentlichen Komponenten des Frameworks.

3.1 Urbane Datenplattform

Die urbane Datenplattform ist der Teil des UDZ, der für die Speicherung von Geodaten zuständig ist, so wie es die DIN SPEC 91357 (2017) beschreibt. Sie besteht aus ver-

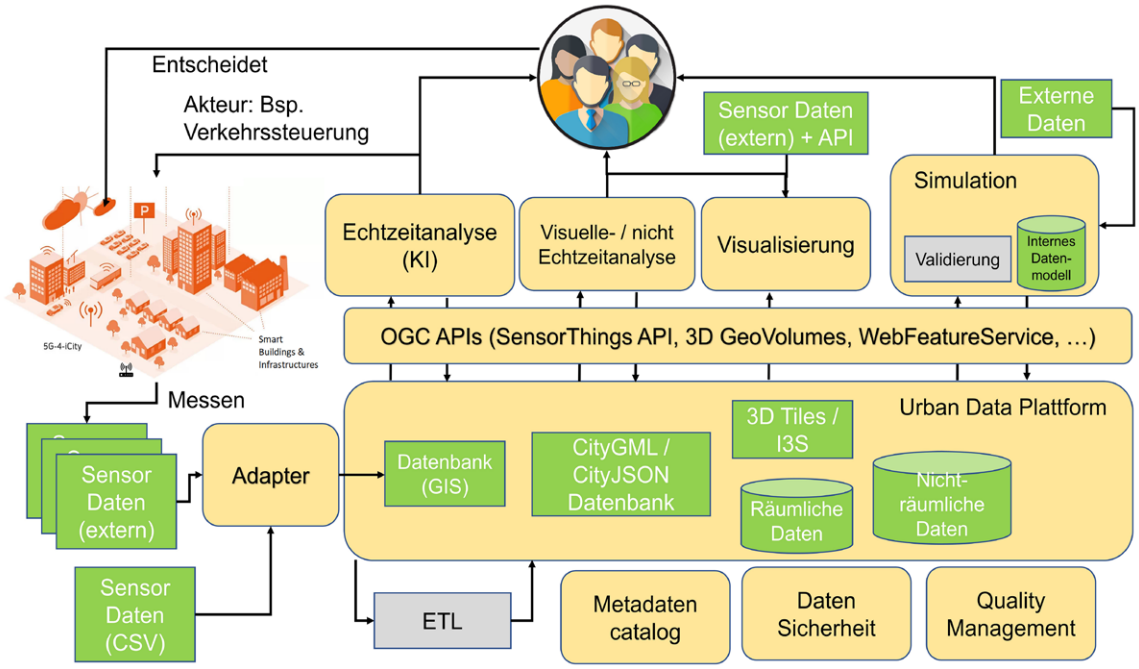


Abb. 3: UDZ-Framework-Design

schiedenen Datenbanken und Datenseen (Speicherung von großen Datenmengen im Rohdatenformat) mit unterschiedlichen Anwendungszwecken, um die (Geo-)Daten aufzunehmen. Die 3D-Modelle werden dabei noch mit Attributen und semantischen Daten erweitert. Diese Attribute werden über eine persistente ID miteinander verknüpft. Falls die Attribute keine geometrische Repräsentation im Modell haben, so kann diese hinzugefügt oder über ein Gruppierungssystem abgebildet werden.

Ein etabliertes Datenformat, um Geodaten zu speichern, ist CityGML. CityGML bietet auch die Möglichkeit, über generische Attribute, generische Objekte und Application Domain Extensions (ADE) das Datenmodell zu erweitern. Das Format verknüpft zudem die geometrischen Informationen mit semantischen Attributen, die zur Interpretation der Geometrie notwendig sind. CityGML wird hier als konzeptionelles Modell eines CIM genutzt. Das Framework lässt offen, welches Encoding verwendet wird. Abhängig vom Anwendungszweck kann ein Encoding in CityJSON oder in XML sinnvoll sein. In den zur Evaluation implementierten Anwendungen wurde jedoch ausschließlich das CityGML XML Encoding verwendet.

Für eine Übersicht über die gespeicherten Daten wird ein Datenkatalog eingesetzt, der Metadaten über die Datensätze verwaltet.

3.2 Sensordaten

Daten, die aus verschiedenen in der Umwelt integrierten Sensoren gesammelt werden, sind Grundlage für Echtzeitanalysen und -simulationen in einem urbanen digitalen Zwilling. Diese Daten sind mit einem Zeitstempel versehen und unterliegen einer dauerhaften Dynamik. Die SensorThings API ist ein OGC-Standard, der die Nutzung von Sensordaten mit Raumbezug unterstützt. Sie basiert auf RESTful Grundlagen und bietet ein standardisiertes Datenmanagement, eine Geodatenintegration sowie Echtzeitverarbeitung an.

In SensorThings werden Sensordaten an sogenannten Things und einem Ort gebunden. Ein Thing kann dabei zum Beispiel ein Gebäude sein, wenn der Sensor für oder an einem Gebäude ist. Es könnte auch an eine Straßenlaterne oder einen Bus gebunden sein. Dieses dynamische System erlaubt es, akkurate Sensordaten flexibel zu speichern. Dabei kann sich der Standort des Sensors auch ändern, falls sich der Sensor an einem sich bewegenden Thing befindet.

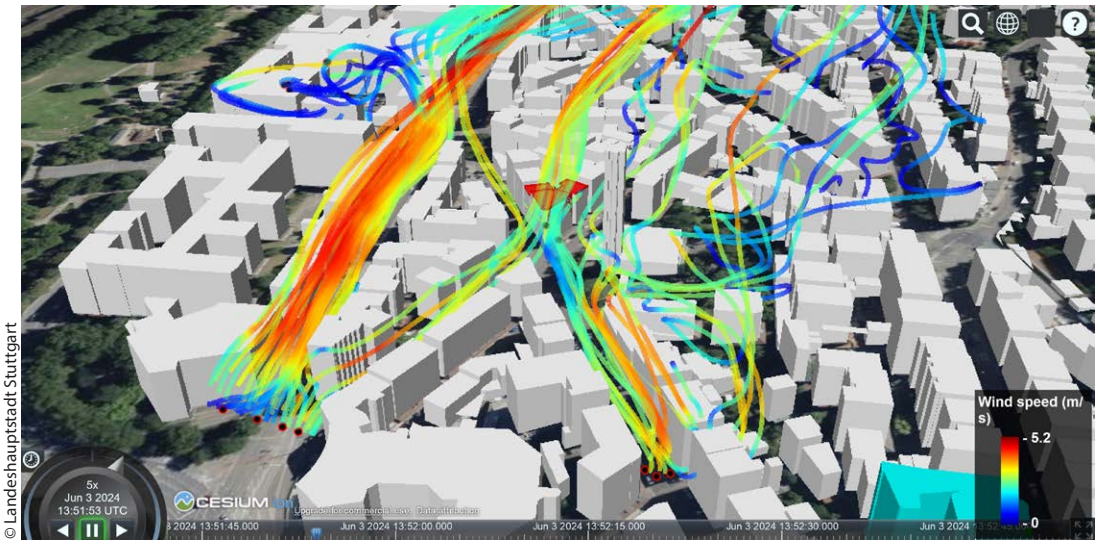
Für die urbane Datenplattform gibt es hier die Unterscheidung zwischen Sensoren, die Teil der Plattform sind, und externen Sensoren, die sich außerhalb der Plattform befinden und mit Hilfe von Schnittstellen in den UDZ eingebunden werden können.

3.3 Prozesse

In einem UDZ werden Daten nicht nur gespeichert und wiedergegeben, sondern es gibt auch verschiedene Prozesse, die innerhalb des Zwillings ablaufen. Manche sind für die Integrität der Plattform notwendig, andere werden für ETL-Prozesse (Extract, Transform, Load) benötigt. Es werden Prozesse für das Qualitätsmanagement gebraucht sowie für Bereitstellung und Aufbereitung der Daten für Simulation und Visualisierung. Auch bei der Datenanalyse und Simulation handelt es sich im weitesten Sinne um Prozesse.

3.4 Analyse & Visualisierung

Das Framework unterstützt eine webbasierte 3D-Visualisierung. Hierzu wird eine progressive Datenübertragung von 3D-Geodaten benötigt, die durch die OGC Communities-Standards 3D Tiles (OGC 22-025r4) und Indexed 3D Scenes (OGC 17-014r9) unterstützt wird. Die vorgeschaltete OGC API 3D GeoVolumes sorgt dabei für einen einheitlichen Zugriff, sodass im Framework sowohl 3D Tiles als



© Landeshauptstadt Stuttgart

Abb. 4: Interaktive Visualisierung einer Windfeldsimulation des Quartiers Stuttgart-Stöckach (3D-Gebäudemodell)

auch Indexed 3D Scenes unterstützt werden. Zusätzliche Daten können über OGC WebFeature Service bzw. OGC API – Features und die SensorThings API eingebunden werden. Diese Elemente werden dann lokal verarbeitet, um eine interaktive 3D-Visualisierung zu erhalten. Mit diesem Prinzip können neben einer interaktiven 3D-Visualisierung auch visuelle Analysen des Datenbestands im Sinne von Visual Analytics (Andrienko et al. 2020) durchgeführt werden. Dies hängt natürlich davon ab, für welchen Zweck und mit welcher Funktionalität die webbasierte Client-Software entwickelt wurde.

Neben der visuellen Analyse, die im webbasierten Client implementiert werden kann, wird also auch eine prädiktive Analyse durch Einbindung von Simulationsanwendungen unterstützt. Die Simulationsumgebung SimStadt (SimStadt 2024) ist direkt in das Framework eingebunden, sodass kleinere Gebiete in Echtzeit über einen API durchgeführt werden können. Für eine rechenintensive Simulation von großen Gebieten und/oder komplexen Simulationen sollten die simulierten Szenarien in einem Offline-Prozess vorberechnet werden, damit eine interaktive Visualisierung möglich bleibt. In diesem Fall werden die Daten von der urbanen Datenplattform heruntergeladen, für die Simulation aufbereitet und die Simulationsergebnisse in die urbane Datenplattform zurückgespielt. Ein Beispiel einer komplexen CFD-Simulation eines Windfeldes für ein Stadtquartier und des entsprechenden Workflows ist in (Deininger 2020) beschrieben (siehe auch Abb. 4).

4 Implementierung

Für die Implementierung des Frameworks wurden grundsätzlich Komponenten genutzt, die als Open-Source-Software zur Verfügung stehen. Allerdings sind nicht für alle Komponenten geeignete Open-Source-Lösungen vorhanden. Einzelne Komponenten wie CityDoctor zum Qualitätsmanagement, die Simulationssoftware SimStadt und deren Anbindung über die OGC API – Processes wurden selbst entwickelt.

Für die Speicherung von CityGML wird die 3DCityDB[1] und die georocket-Datenbank[2] verwendet. Die Datenbank für die 3DCityDB ist eine Postgres[3]-Datenbank mit der PostGIS[4]-Erweiterung. Für die georocket-Datenbank wird eine NoSQL MongoDB[5] verwendet. In der Implementierung wurden dabei die Attribute, die mit Geodaten verbunden sind, in einer separaten Datenbank gespeichert. Da die Attribute komplexe Datenstrukturen sein können, wurde auch hier eine NoSQL MongoDB-Datenbank eingesetzt, die eine JSON-Repräsentation der Daten speichert und wiedergeben kann. Dies ist insbesondere bei der Verarbeitung in JavaScript bei der Visualisierung sehr einfach zu verwenden. Die Assoziierung der Attribute erfolgt über die GML-ID der CityGML-Daten. Für Datensätze, die nicht an einzelnen Gebäuden oder anderen Objekten gebunden sind, kann entweder ein neues GenericCityObject erstellt werden, dessen Geometrie zu dem Datensatz passt, oder es kann eine Gruppierung von Objekten in CityGML erstellt werden. Beispielsweise kann es Gebäudestatistiken auf Postleitzahlebene geben, die entweder dem Gebiet oder einer Gruppe von Gebäuden, die das Gebiet umfasst, zugeordnet werden kann. Für das Gebiet kann deshalb eine 2D-Geometrie in einem GenericCityObject erstellt werden, der dann die Attribute zugeordnet werden. Eine Aggregierung von Daten funktioniert analog. Es können Datensätze von einzelnen Objekten aggregiert werden und zu einem GenericCityObject oder einer Gruppe assoziiert werden.

Für das Qualitätsmanagement kommt das Programm CityDoctor[6] zum Einsatz, welches die CityGML-Modelle der Plattform nach geometrischen und semantischen Anforderungen prüft und einen Prüfbericht erstellt.

Für die Visualisierung werden die Bibliotheken CesiumJS oder ArcGIS Maps SDK for JavaScript für die Darstellung von georeferenzierten 3D-Modellen im Browser genutzt. Für die Speicherung und Bereitstellung von Sensordaten wird ein FROST(FRaunhofer Opensource SensorThings)-Server[7] eingesetzt, der eine Open-Source-Implementierung der OGC-Schnittstelle SensorThings bietet. Die Datenverwaltung übernimmt auch hier eine Postgres-Datenbank mit der PostGIS-Erweiterung. Als ETL-Komponente kommt FME[8] zum Einsatz, das eine große Bandbreite an Formaten lesen, konvertieren und wieder schreiben kann.

Für die Metadatenhaltung zu Daten der Datenplattform, externer Datenquellen und auch Prozessen wird eine CKAN-Katalog eingesetzt (siehe auch Abb. 5).

4.1 SimStadt & SimStadt API

Für die Simulation wird die Simulationsplattform SimStadt verwendet. SimStadt ist als Programm frei verfügbar. Im Kontext der Datenplattform ist jedoch eine Web-Schnittstelle notwendig, wofür die SimStadt API aufgebaut wurde. Sie verwendet intern die Simulationsroutinen der Plattform, jedoch ohne dafür eine grafische Oberfläche zu benötigen. Stattdessen läuft die Konfiguration über eine HTTP-Schnittstelle ab. Dafür wird ein Gebiet in Form einer Bounding Box zusammen mit einer Liste von Attributen, die SimStadt simulieren soll, übergeben. Außerdem kann der Simulationsprozess mit Parametern konfiguriert werden.

1 3DCityDB Version 4.4
2 Georocket-Datenbank Version 1.3.0
3 Postgres Version 12.18
4 PostGIS-Erweiterung Version 3.0.0
5 MongoDB Version 6.0.5
6 CityDoctor Version 3.13.0
7 https://github.com/FraunhoferIOSB/FROST-Server
8 https://fme.safe.com/platform/

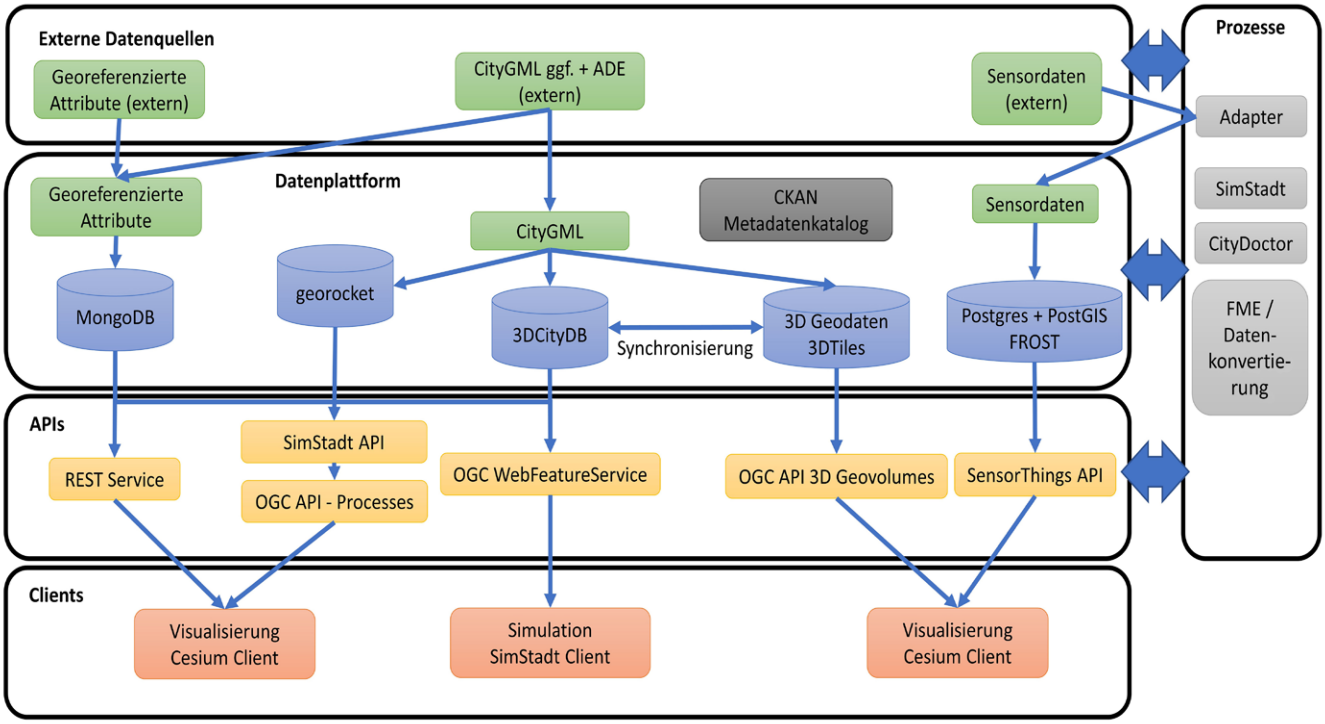


Abb. 5: Übersicht über die eingesetzten Komponenten und Schnittstellen

Der Prozess auf der Serverseite holt sich von einer georocket-Datenbank für das übergebene Gebiet alle Gebäude und andere Features. Mit den Daten wird eine SimStadt-Simulation durchgeführt, die speziell auf die angeforderten Attribute zugeschnitten ist. Sobald der Simulationsprozess abgeschlossen ist, werden die berechneten Daten den GML_IDs der einzelnen Objekte zugeordnet und zurückgegeben.

Die API ist dabei zusätzlich in die OGC API – Processes eingebunden. Die Processes API bindet hierbei die SimStadt API in eine standardisierte Web API ein und fügt die entsprechenden Metadaten zur Nutzung der Schnittstelle hinzu. Zusätzlich werden Ergebnisse von vergangenen Simulationen über einen Cache zwischengespeichert, sodass gleiche Anfragen nicht wiederholt simuliert werden, sondern das Ergebnis direkt zurückgegeben werden kann. Es wird auch ermöglicht, die Simulation asynchron auszuführen. Dies ist notwendig, um insbesondere bei länger andauernden Simulationen die Interaktivität des Systems zu gewährleisten.

Neben der Nutzung der API kann SimStadt auch über eine grafische Benutzungsoberfläche verwendet werden. Dazu wurde eine Schnittstelle entwickelt, die CityGML mit Attributen anreichert und zur Verfügung stellt. Der Fokus liegt dabei auf den im CityGML-Standard festgelegten Attributen wie dem Baujahr, die für bestimmte Simulationen notwendig sind.

5 Evaluation

Dieses Framework wurde in einer Reihe von Fallstudien ausgiebig getestet, die alle auf einer ähnlichen Struktur aufbauen.

5.1 EnSys-LE

In dem Projekt EnSys-LE wurden Simulationsdaten aus SimStadt mit Gebäudedaten für die Landkreise Dithmarschen (28.054 Gebäude), Ludwigsburg (80.996 Gebäude) und Ilm-Kreis (18.118 Gebäude) visualisiert. Es handelt sich um einen länderübergreifenden Datenbestand, der das 3D-Gebäudemodell der ZSHH über das BKG verwendet und mit Baujahren aus einer externen Datenquelle kombiniert. Die Simulationsdaten sind vorberechnet und werden zusammen mit der Geometrie via 3D-Tiles in Cesium dargestellt (Santhanavanich et al. 2022). Zu den simulierten Daten gehören der Wärmebedarf zusammen mit dem spezifischen Wärmebedarf sowie das PV-Potenzial der Gebäude und der Strombedarf. Die Gebäude werden je nach Ergebnis eingefärbt und bieten damit eine räumlich visuelle Analyse (siehe auch Abb. 6).

In diesem UDZ wurden die Komponenten 3D-Tiles für die Visualisierung, SimStadt für die Simulation der Energiedaten und CityDoctor für das Qualitätsmanagement evaluiert. Hinzu kommt die Verarbeitung der Daten mit FME, um die CityGML-Daten mit Baujahren zu erweitern.

Abb. 6: EnSys-LE Dithmarschen Wärmebedarf von Gebäuden

5.2 Fallstudie Montreal im OGC Testbed 18

Im OGC Testbed 18 (St. Hilaire et al. 2023) wurde die Interoperabilität zwischen 3D-Stadtmodellen und Gebäudeenergiedatensätzen im Zusammenhang mit den OGC APIs geprüft. Unter anderem wurde die Verbindung von SimStadt-Simulationsdaten mit den 3D-Gebäudedaten mit Hilfe der EnergyADE untersucht. Die EnergyADE ermöglicht die Speicherung der simulierten Daten in den CityGML-Dateien, um sie später zu visualisieren oder weiterzuverarbeiten.

In diesem Projekt wurde auch die SimStadt API über die OGC API – Processes verbunden (siehe auch Abb. 7). Die SimStadt API berechnet die Ergebnisse dabei nach Anfrage der Nutzer und ist nicht vorberechnet. Dadurch ergeben sich personalisierte Einstellungsmöglichkeiten der Simulation je nach Anwendungszweck der Nutzer im Web-Client.

Auch die Bereitstellung der Simulationsdaten über die SensorThings API wurde untersucht. Dabei sollten Simulationen als Sensordaten eines möglichen Zukunftsszenarios interpretiert und damit konzeptionell wie Messdaten behandelt werden.

5.4 Fallstudie Munakata

In Munakata wurde in Kooperation mit Partnern aus Japan ein UDZ-Prototyp für die Stadt Munakata erstellt. Dabei kam nicht nur die Simulationsplattform SimStadt

Abb. 7: Wärmebedarfssimulation über die OGC API – Processes verbunden mit der SimStadt API

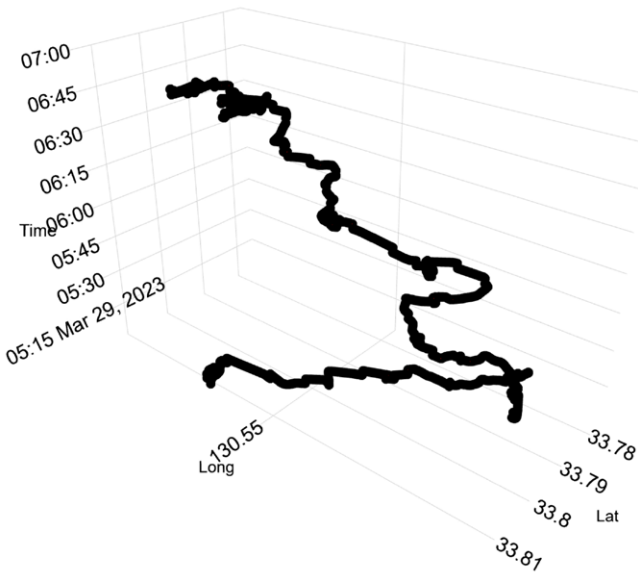


Abb. 8: Eine Busroute, dargestellt als Raum-Zeit-Würfel

zum Einsatz, sondern es wurden auch Sensordaten und eine Busroute in die Visualisierung integriert (siehe auch Abb. 9).

Die Datenquellen sind ein CityGML-Modell, erweitert mit einer Urban Planning ADE sowie Sensordaten über die externe Plattform Ubibot. Die für die Simulation benötigten Baujahre sind in der Urban Planning ADE als Baujahresklassen vorhanden. Außerdem wurden Daten zu Busrouten zur Verfügung gestellt.

Diese Routen wurden von einem Bus aufgenommen und in eine Datenplattform integriert. In der Client-Anwendung kann diese Busroute analysiert werden, um zu sehen, wie, wann und wo der Bus gefahren ist. Anhand der Daten der Busroute wurde die Datenbereitstellung über die SensorThings API und die OGC API – MovingFeatures untersucht. Über beide Schnittstellen können Sensordaten an beweglichen Objekten gebunden werden. Die Daten werden auch in einem Raum-Zeit-Würfel dargestellt, der eine interessante Möglichkeit zur visuellen Analyse bietet. In Abb. 8 ist ein Raum-Zeit-Würfel zu sehen. Dabei handelt es sich um eine 3D-Darstellung, bei der die X- und Y-Achse den Längen- und Breitengrad der Position des Busses abbilden. Die Z-Achse stellt die Zeitkomponente dar, sodass eine Route in Raum und Zeit dargestellt werden kann.

Zudem werden drei Sensoren der Firma Ubibot eingesetzt, die Daten über Temperatur, Luftfeuchtigkeit und Helligkeit aufnehmen. Diese Daten werden von den Sensoren an eine proprietäre Plattform der Firma übertragen. Die externe Plattform hat keine eigene SensorThings API-Schnittstelle und wurde deshalb über einen Adapter in unsere Datenplattform integriert. Die übrigen Komponenten können dadurch wie gewohnt über die SensorThings API auf die Sensoren zugreifen. Der Adapter dient als Fassade für die externe Sensordatenplattform.

6 Diskussion

Das vorgestellte Framework bildet eine Blaupause, um einen UDZ aufzubauen. Dabei werden passende Komponenten zusammengefasst und mit einer standardisierten Schnittstelle verbunden. Die einzelnen verbundenen Komponenten sind dabei austauschbar in ihrer Implementierung, sofern sie dieselbe Schnittstelle unterstützen. Als Schnittstellen werden OGC-Standards genutzt. Es zeigt zudem, dass die Kodierung der Daten nachrangig ist, denn solange die Schnittstelle bedient werden kann, ist das Datenformat nur in der technischen Betrachtung wichtig. Die Funktion ist davon unabhängig.

Um externe Datenquellen einzubinden, die keine standardisierte Schnittstelle verwenden, muss eine Datenkonvertierung stattfinden. Dazu muss entweder die Schnittstelle mit Hilfe eines Adapters zur Laufzeit übersetzt werden

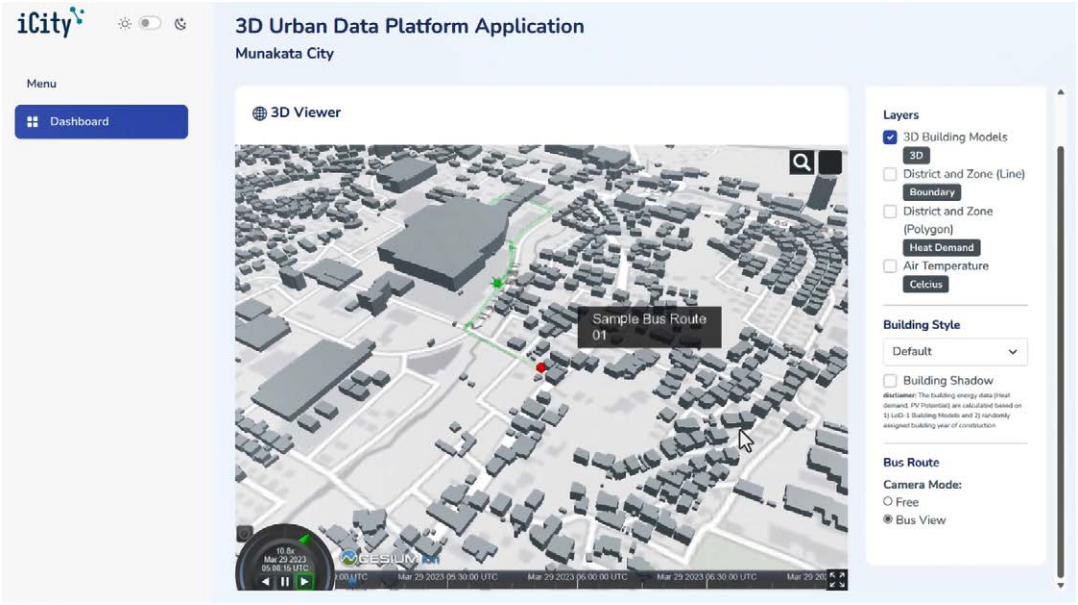


Abb. 9: Eine Busroute, dargestellt mit den 3D-Gebäuden

oder die Daten müssen in einem geeigneten Format zwischengespeichert werden. Zum Beispiel gibt es viele Sensoren, die über das Internet verfügbar sind. Diese Sensordaten sind aber selten über die OGC – SensorThings API erreichbar, sondern haben entweder eigene Schnittstellen, verwenden andere Protokolle oder führen keine Historie über ihre Daten. Damit diese Sensoren in einem UDZ erfolgreich eingebunden werden können, muss deshalb eine Verarbeitung vorher stattfinden, um die Daten einfacher nutzbar zu machen.

In der vorgestellten Implementierung ist der CKAN-Katalog noch lose integriert. Durch eine Integration von OGC-Schnittstellen könnte eine vereinfachte Nutzung der Datensätze über APIs direkt aus CKAN heraus ermöglicht werden.

7 Fazit und Ausblick

Es konnte gezeigt werden, dass es mit vorhandenen offenen Technologien und Komponenten möglich ist, unterschiedliche urbane digitale Zwillinge zu erstellen, die auf einen bestimmten Anwendungszweck zugeschnitten sind. In der Fallstudie EnSys-Le wurde auch gezeigt, dass das Framework mit großen Datenbeständen – hier ein Modell von drei Landkreisen – umgehen kann.

Bei der Implementierung gibt es noch Verbesserungsmöglichkeiten. Momentan ist die Simulationsschnittstelle SimStadt API eng an die georocket-Datenbank gebunden. Hier sollte die SimStadt API in Zukunft über eine OGC API – Features oder eine WebFeatureService-Schnittstelle auf die für die Simulation notwendigen Daten zugreifen können. Dadurch wäre die Datenhaltung austauschbar.

Danksagung

Die in diesem Beitrag vorgestellten Arbeiten sind in der vom Bundesministerium für Bildung und Forschung im Rahmen des Förderprogramms Forschung an Fachhochschulen geförderten Projekt-Partnerschaft iCity, insbesondere in den Projekten ICT4iCity (FKZ 13FH9I02IA), UDigiT4iCity (FKZ 13FH9I06IA) und DiaOpt4iCity (FKZ 13FH9I10IA) entstanden. Für diese Förderung bedanken wir uns beim BMBF.

Literatur

Andienko, N., Andrienko, G., Fuchs, G., Slingsby, A., Turkay, C., Wrobel, S. (2020): Visual Analytics for Data Scientists. Springer Nature Schweiz. DOI: https://doi.org/10.1007/978-3-030-56146-8.

Deininger, M. E., Grün, M., Piepereit, R., Schneider, S., Santhanavanich, T., Coors, V., Voß, U. (2020): A Continuous, Semi-Automated Workflow: From 3D City Models with Geometric Optimization and CFD Simulations to Visualization of Wind in an Urban Environment. ISPRS Int. J. Geo-Inf. 2020, 9, 657.

DIN SPEC 91357 (2017): Referenzarchitekturmodell Offene Urbane Plattform (OUP).

Donaubauer, A., Knezevic, M., Willenborg, B., Bobinger, S., Morich, L. (2023): Leitfaden – Urbaner Digitaler Zwilling nach der Methodik der SDDI (Version 1.2). TU München. https://mediatum.ub.tum.de/doc/1725270/document.pdf, letzter Zugriff 28.4.2024.

GI (2024): Digitaler Zwilling. Gesellschaft für Informatik e. V. https://gi.de/informatiklexikon/digitaler-zwilling, letzter Zugriff 28.4.2024.

Kalasatama (2019): The Kalasatama Digital Twins Project. www.hel.fi/static/liitteet-2019/Kaupunginkanslia/Helsinki3D_Kalasatama_Digital_Twins.pdf, letzter Zugriff 28.4.2024.

LHS Stuttgart (2024): Digitaler Zwilling für Mobilität und Umwelt | Landeshauptstadt Stuttgart. www.stuttgart.de/leben/bauen/geoportal/digitaler-zwilling-mobilitaet-und-umwelt, letzter Zugriff 28.4.2024.

Oestereich, B. (2004): Objekt-orientierte Softwareentwicklung – Analyse und Design mit der UML 2.0. 6. Auflage, Oldenbourg Verlag, München Wien, ISBN 3-486-27266-7.

OGC 17-014r9 (2023): OGC Indexed 3D Scene Layer (I3S) and Scene Layer Package (*.slpk) Format Community Standard, Version 1.3.

OGC 22-025r4 (2023): 3D Tiles Specification, Version 1.1.

Ostrau, S. (2024): Deutschland im digitalen Umbruch – Wege zur stärkeren Etablierung der GDI-DE. In: zfv – Zeitschrift für Geodäsie, Geoinformation und Landmanagement, Heft 1/2024, 149. Jg., 52–59. DOI: 10.12902/zfv-0457-2023.

Raes, L., Michiels, Adolphi, T., Tampere, C., Dalianis, A., McAleer, S., Kogut, P. (2022): DUET: A Framework for Building Interoperable and Trusted Digital Twins of Smart Cities. In: IEEE Internet Computing, vol. 26, no. 3, 43–50, 1 May-June 2022. DOI: 10.1109/MIC.2021.3060962.

Santhanavanich, T., Padsala, R., Würstle, P., Coors, V. (2022): The spatial data infrastructure of an urban digital twin in the building energy domain using OGC standards. ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci., X-4/W2-2022, 249–256. DOI: 10.5194/isprs-annals-X-4-W2-2022-249-2022.

Schubbe, N., Boedecker, M., Moshrefzadeh, M., Dietrich, J. Mohl, M. Brink, M., Reinecke, N., Tegtmeyer, S., Gras, P. (2023): Urbane Digitale Zwillinge als Baukastensystem: Ein Konzept aus dem Projekt Connected Urban Twins (CUT). In: zfv – Zeitschrift für Geodäsie, Geoinformation und Landmanagement, Heft 1/2023, 148. Jg., 14–23. DOI: 10.12902/zfv-0417-2022.

SimStadt (2024): https://simstadt.hft-stuttgart.de, letzter Zugriff 28.4.2024.

St. Hilaire, L., Brookson, A., Jacovella-St-Louis, J., Dion, P., Coors, V., Santhanavanich, T., Padsala, R., Dabirian, S., Saad, M. M., Eicker, U., Rossknecht, M., Hodgson, T., Bovio, S., Portele, C., Morton, W., Mottaghi, A. (2023): Testbed-18: Building Energy Data Interoperability Engineering Report.

Kontakt

Prof. Dr. Volker Coors | Matthias Betz | Thunyathep Santhanavanich
Hochschule für Technik Stuttgart
Schellingstraße 24, 70174 Stuttgart
volker.coors@hft-stuttgart.de
matthias.betz@hft-stuttgart.de
thunyathep.santhanavanich@hft-stuttgart.de

Dieser Beitrag ist auch digital verfügbar unter www.geodaesie.info.