Hochschule für Technik und
Wirtschaft Dresden
University of Applied Sciences

Hochschule für Technik und Wirtschaft
Dresden

Masterarbeit

Multimodel-Simulationen und Ergebnisanalyse mittels Business Intelligence zur Optimierung dynamischer Strompreise

Marinovic, Tom

Betreuer:
Prof. Dr.-Ing. Thomas Wiedemann
Zweit-Gutachter:
Prof. Dr.-Ing. Axel Toll

Bearbeitungszeitraum vom:
22.02.2025 bis 22.07.2025

Abgabe am:
18. Juli 2025

Inhaltsverzeichnis

1 Einleitung **1**
1.1 Motivation 1
1.2 Forschungsfrage & Methodik 2

2 Theoretische Vorbereitung **4**
2.1 Dynamische Strompreise 4
2.2 Themen Relevanz & Abgrenzung 6
2.3 Diskrete Simulation 8
2.4 Eingesetzte technische Lösungen 10
2.5 Multimodel Validierung 12
2.6 Bestehendes DynPlus Preismodell 15

3 Planung **17**
3.1 Systementwurf und Modellierung 17
3.2 Schnittstellendefinition 19
3.3 Datenstrukturen & ERM 20
3.4 Dynamische Datendarstellung 23

4 Entwicklung **26**
4.1 Umsetzung Systemstruktur 26
4.2 Implementation Validierung 29
4.3 Aufbau Data Warehouse 31
4.4 Einrichten der BI Funktionen 33
4.5 Datenauswertung 35
4.6 Systemtest 36

5 Diskussion **38**
5.1 Bewertung des Modells als Entscheidungsgrundlage 38
5.2 Technische Alternativen 39
5.3 Robustheit und Anpassbarkeit 41
5.4 Einschränkungen (Limitations) 42

6 Abschluss **44**
6.1 Fazit der Arbeit 44
6.2 Vergleich zur Zielstellung 45
6.3 Ausblick & Folgeentwicklung 46

7 Literaturverzeichnis **47**

8 Anlagen **50**

9 Selbstständigkeitserklärung **56**

Abkürzungen

API Application Programming Interface. 6, 26

BI Business Intelligence. 2, 3, 23, 45, 46

DBMS Datenbank Management System. 39

DWH Data Warehouse. 17, 46

ERM Entity-Relationship-Modell. 20, 22

ETL Extract-Transform-Load. 21, 28, 37

KPI Key Performance Indicator. 18

MDX Multidimensional Expressions. 32, 33

MMF Multimodel Framework. 12, 14

OLTP Online Transaction Processing. 17, 20, 26, 46

ROLAP Relational Online Analytical Processing. 18

SQL Structured Query Language. 10, 20

UML Unified Modeling Language. 19

Glossar

Business Intelligence Business Intelligence ist die Analyse von Geschäftsdaten, um aus ihnen verwertbare Erkenntnisse zu erlangen, welche Unternehmen helfen fundierte Entscheidungen zu treffen. 2, 10, 17, 33

Cube (OLAP-Würfel) Ein mehrdimensionales Datenmodell, das zur Analyse großer Datenmengen entlang verschiedener Dimensionen (z.B. Zeit, Modell, Szenario) dient.. 17

Data Vault Ein datenbankgestütztes Modellierungskonzept für Data Warehouses, das auf Hubs, Links und Satelliten basiert. Es ermöglicht hohe Flexibilität, Historisierung und Skalierbarkeit.. 41

Data Warehouse Wörtlich übersetzt Datenlager, bezeichnet eine Datenbank zu Analyszwecken, welche aus verschieden Quellen zusammengeführt wird. 10, 17

diskrete Simulation Die Nachbildung eines Modells mittels Rechentechnik mit sprunghaften Wertveränderungen. 3, 8

DynPlus Bezeichung für ein alternatives dynamisches Strompreismodel, mit Abgabendecklung auf 200%, erstmals vorgestellt in [Wie24]. 2, 5

Lock-In Effekt Beschreibt die Abhängigkeit eines Kunden von einem Anbieter, Produkt oder System, wobei ein Wechsel mit hohen Kosten oder Aufwand verbunden ist. Dies kann technologische, wirtschaftliche oder vertragliche Ursachen haben. 6

PivotTable Ein interaktives Werkzeug zur Datenanalyse in Excel, das auf OLAP-Cubes zugreifen kann.. 35

Smart Meter Ein intelligenter Stromzähler, der den Energieverbrauch in Echtzeit misst und überträgt. Er ermöglicht eine zeitgenaue Abrechnung und ist Voraussetzung für dynamische Stromtarife.. 6

staging Area Ein temporärer Speicherbereich, in dem Daten vor der Weiterverarbeitung oder Analyse zwischengespeichert werden.. 24

Abbildungsverzeichnis

1 Dynamische Strompreise in Cent (Y-Achse) am 09.03.25 in Dresden . 1
2 Aufteilung des Strompreises am 08.03.2024 (entnommen [Wie24] S.4 Abb. 5 und angepasst) 5
3 Klassifikation von Simulationsmodellen, nach [MM89] Abb. 1 8
4 Übersicht der SQL Server Lizenzierung entnommen dem SQL Server Lizenz Datenblatt 11
5 Zerlegung des Operation Space nach MMF, entnommen [AF17] S.1150 Fig.2 13
6 Zusammenführen lokaler Modelle, entnommen [AF17] S.1151 Fig.3 . . 14
7 Beispiel für die Preisverschiebung 15
8 Darstellung der geplanten Gesamtstruktur 17
9 Komponentendiagramm der Gesamtstruktur 19
10 ERM der OLTP Datenbank 20
11 ERM der DWH Datenbank 22
12 Darstellung Systemaufbau 24
13 Verteilung der generierten Steuerwerte 27
14 Validierung Ergebnisbeispiel (angepasste Darstellung) 30
15 Data Warehouse Cube Darstellung 32
16 Vergleich Historische Strompreise mit *Dyn-Plus* Simulations-ergebnissen (angepasste Darstellung) 36

1 Einleitung

1.1 Motivation

Dunkelflauten waren bis vor wenigen Jahren vielen Menschen noch kein Begriff, doch sind sie inzwischen in den Schlagzeilen der Presse zu finden und schüren dort nicht selten Ängste vor explodierenden Strompreisen und einer unsicheren Stromversorgung in Deutschland [Mic25]. Dunkelflauten bezeichnen Zeiträume von Windstille und fehlender Sonneneinstrahlung, wodurch die Gewinnung erneuerbarer Energien aus diesen Quellen nur sehr eingeschränkt möglich ist. Obwohl sie in der Presse teilweise überspitzt dargestellt werden, stellen Dunkelflauten durchaus ein Problem beim fortschreitenden Ausbau erneuerbarer Energien dar. Beispielsweise können sie dafür sorgen, dass sich der Strompreis an der Strombörse - im Zeitraum einer Dunkelflaute - gegenüber dem Durchschnittspreis versiebenfacht [Bun24a]. Dies stellt nicht nur ein massives Problem für Unternehmen mit hohem Strombedarf dar, sondern auch für private oder geschäftliche Abnehmer, die einen dynamischen Strompreis (im Folgenden auch dynPreis) beziehen. Ein dynamischer Strompreis ist abhängig von den Preisen an der Strombörse, wird stets 24 Stunden im Voraus festgelegt und kann von verschiedenen Anbietern wie z.B. Tibber [Zei25] bezogen werden. Durch den fortwährenden Ausbau erneuerbarer Energien ergeben sich Zeitpunkte (besonders um die Mittagszeit), in denen deutlich mehr Strom verfügbar ist und dementsprechend günstiger angeboten werden kann. Dies kommt den Nutzern von dynamischen Strompreisen jedoch nur bedingt zugute; stehen diesem Phänomen doch die Dunkelflauten gegenüber, welche umso mehr ein Problem darstellen da sie die Preise in die Höhe treiben. Zusätzlich kann der Endverbraucher die niedrigen Stromerzeugungskosten der Mittagszeit aufgrund von Abgaben und Steuern, die stets entrichtet werden müssen, kaum ausnutzen, erkennbar auch in folgender Grafik.

Abbildung 1: Dynamische Strompreise in Cent (Y-Achse) am 09.03.25 in Dresden

Wie zu erkennen ist, sinkt der Strompreis zur Mittagszeit auf bis zu -0,2 ct\kWh; die Abgaben belaufen sich jedoch auf 22,21 ct\kWh. Durch dieses Abgabeverhalten bietet der dynamische Strompreis momentan nur einen sehr geringen Vorteil, wenn nicht sogar einen Nachteil, gegenüber einem klassischen fixen Strompreis. Dieses Problem kann jedoch durch eine Abgabenverlagerung behoben werden, gezeigt durch die Simulation eines alternativen Strompreis Modells, vorgestellt in [Wie24].

1.2 Forschungsfrage & Methodik

Fast die Hälfte des deutschen Strombedarfs wird derzeit von erneuerbaren Energien gedeckt, mit steigender Tendenz [Bun25a]. Davon haben Windkraft gefolgt von Photovoltaik die größten Anteile, was zu Schwankungen in der Stromproduktion über den Tag hinweg führt. Dies bringt jedoch mehrere Nachteile mit sich, nicht zuletzt, weil die Probleme und Auswirkungen von Dunkelflauten zukünftig nur noch stärker werden. Damit diese potenziell starken Schwankungen ausgeglichen werden können, muss auch ein fester Strompreis entsprechend hoch angesetzt werden. Damit ein Strompreisanbieter wirtschaftlich bleibt, müssen die Extrempreise der Dunkelflauten auf das ganze Jahr verteilt werden, ohne Sicherheit, ob diese überhaupt eintreten. Zudem produzieren erneuerbare Energien über den Tag hinweg nicht gleichmäßig Strom, stattdessen kommt es zeitweise zur Stromüberproduktion. Welche in den sehr niedrigen Stromerzeugungskosten zur Mittagszeit resultiert, siehe Abbildung 1. Ein Endverbraucher mit einem dynamischen Stromtarif, kann Strom zu diesen niedrigen Preise beziehen. Aber wie zuvor erwähnt, sind die dynamische Strompreise im momentan existierenden Preismodell nur beschränkt wirtschaftlich sinnvoll. Das Stromnetz wiederum wird durch diese Überproduktion zusätzlich belastet, weswegen eine Veränderung des regulären Verbrauchsmusters von Vorteil wäre. Das reguläre Verbrauchsmuster bezieht sich hier auf private Endverbraucher, die im Durchschnitt zur Mittagszeit nicht zu Hause sind und abends zu ähnlichen Zeiten heimkehren. Diese Veränderung ließe sich durch ein wirtschaftlich sinnvolles, dynamisches Strompreismodell wie beispielsweise *DynPlus*, vorgeschlagen in [Wie24], erreichen. Ziel dieser Arbeit ist es, dieses und weitere alternative Strompreismodelle zu vergleichen und zu bewerten. Dabei liegt der Fokus auf dem parallelen Vergleich verschieden strukturierter Strompreissimulationen in einem geschlossenen System, das die Möglichkeit bietet, mittels Business Intelligence (im Folgenden BI)-Werkzeugen ausgewertet zu werden. Die Forschungsfrage der Arbeit lässt sich somit folgendermaßen formulieren: **„Wie lassen sich verschieden strukturierte Simulationsmodelle zielführend und wirtschaftlich effektiv miteinander vergleichen.“**

Dabei ist zu spezifizieren, dass es sich bei allen Modellen um eine Modellierung des selben Problems, aber mit unterschiedlichen Eigenschaften je Modell, handelt. Dies bedeutet, dass alle Modelle die diskrete Simulation dynamischer Strompreise zum Ziel haben, ihre Ergebnisse, Zwischenschritte, Zeiteinheiten oder eigenen Variablen jedoch unterschiedlich sein können. Die Komplexität entsteht vorwiegend beim Harmonisieren der unterschiedlichen Eingangs- und Ergebnisstrukturen. Da es das Ziel ist, einem „Entscheider“ mittels BI-Werkzeugen eine vergleichende Auswertung zu ermöglichen, muss die Systemstruktur eine Vereinheitlichung durchführen. Dabei muss für jeden Wert der Rückschluss auf das erzeugende Modell gegeben sein, damit unterschiedliche Simulationsergebnisse nachvollzogen werden können. Eine detaillierte Beschreibung der Problemhintergründe findet sich im folgenden Kapitel *Theoretische Grundlagen*. Die verwendete Methodik zur Beantwortung dieser Forschungsfrage ist eine Kombination aus systematischer Literaturanalyse und eigenständiger Entwicklung. Ziel dieser Vorgehensweise ist es, auf den theoretischen Grundlagen für den Multimodelvergleich aufbauend, dass noch wenig erforschte Gebiet der dynamischen Strompreise durch Eigenentwicklung besser erfassen zu können. Im ersten Schritt wurden dafür verschiedene Vorgehensweisen für den Vergleich von unterschiedlichen Simulationsmodellen aus der bestehenden Forschung untersucht. Nach der Auswahl eines geeigneten Ansatzes konnte dieser dann auf die Forschungsfrage angewandt und mittels Eigenentwicklung implementiert werden. Wie zuvor erwähnt, ist die bestehende wissenschaftliche Basis zu dynamischen Strompreisen noch sehr flach, daher wurden auch Ansätze aus anderen Disziplinen als der Informatik betrachtet. Hierbei waren vor allem Arbeiten aus dem Bereich der Klimawissenschaften und Geodäsie besonders hilfreich, da der Aspekt des Multimodel-Vergleichs in diesen Feldern ein häufiges Thema ist. Die Kombination aus beiden methodischen Ansätzen ermöglichte somit eine theoretisch fundierte und gleichzeitig praxisrelevante Vorgehensweise beim Beantworten der Forschungsfrage. Zusätzlich sollte auch die Darstellung und Auswertung der entstehenden komplexen Datenstrukturen mit verschiedenen Mitteln abgewogen werden. Zu diesem Ziel wurde, in Zusammenarbeit mit einer gleichzeitig in Bearbeitung befindlichen Diplomarbeit, die dynamisch anpassbare Darstellung eines Datenabschnitts auf einer Website implementiert. Diese Diplomarbeit wird von Lukas Möbius erstellt und trägt den Titel "*Interaktive Webanwendung zur simulationsgestützten Analyse und Bewertung von dynamischen Strompreisen und Vergleich alternativer Abgabensysteme*"[Möb25].

2 Theoretische Vorbereitung

2.1 Dynamische Strompreise

Bevor die technischen Aspekte dieser Forschungsfrage beleuchtet werden können, sollen einige theoretische Grundlagen zum besseren Verständnis der Thematik aufgearbeitet werden. Stromlieferanten wurden erstmalig durch das Energiewirtschaftsgesetz § 41a, in Kraft getreten am 29.12.2023, verpflichtet, ihren Endkunden dynamische Strompreise anzubieten [Jus23]. Paragraph § 41a Abs. 2 Satz 1 EnWG schränkt dies auf Strompreislieferanten mit mindestens 100.000 Endabnehmern ein und Satz 3 macht dies ab dem 1. Januar 2025 verpflichtend. Ein dynamischer Strompreis im Sinne des Gesetzes und dieser Arbeit ist ein last- und/oder tageszeitabhängiger Preis für den Endabnehmer. Endabnehmer sind Privatpersonen oder Unternehmen, die den angebotenen Strom konsumieren. Es existieren Modelle, die das Wiedereinspeisen von erzeugtem oder gespeichertem Strom in das Stromnetz vergüten. Diese Modelle werden im Zuge dieser Arbeit nicht betrachtet, da für die Betrachtung davon ausgegangen wird, dass ein Endabnehmer sämtlichen Strom, den er bezieht, auch konsumiert. Dabei ist es möglich, dass er den Strom zeitweise zwischenspeichert, um seinen Verbrauch zu optimieren. Eine Verbrauchsoptimierung bedeutet, dass Strom zu niedrigen Preisen bezogen und während Hochpreiszeiten verbraucht wird.
Ein dynamischer Strompreis setzt sich mindestens aus den Kosten für die Energiebeschaffung, dem Netzentgelt und den steuerlichen Abgaben zusammen. Des Weiteren können noch Abgaben und Umlagen erhoben werden, wie zum Beispiel Konzessionsabgaben oder Offshore-Netzumlagen. Die Kosten für die Energiebeschaffung stellen Erzeugung oder Einkauf, sowie Gewinnmargen da. Die steuerlichen Abgaben wiederum setzen sich aus Umsatzsteuer und Stromsteuer zusammen. [Bun24b].
Die Kosten für die Energiebeschaffung entstehen nach dem *Mail and Order* Prinzip, durch Angebot und Nachfrage an einem entsprechenden Handelsplatz. Dies ist beispielsweise die europäische Strombörse EPEX SPOT, deren Daten auch im Zuge dieser Arbeit verwendet werden.
In der Quelle [Wie24] wird der Begriff *Energiebeschaffung* mit dem Nettopreis gleichgesetzt. Dies ist zwar nicht korrekt im Sinne der Fachterminologie aber bietet dem fachfremden Leser ein leichteres Verständnis. Daher werden in dieser Arbeit an ausgewählten Stellen oder Zitaten der Begriff Nettopreis für die Kosten der Energiebeschaffung und der Begriff Bruttopreis für den Strompreis verwendet.
Im Zuge des Föderalismus kann die Höhe der Abgaben und Umlagen je nach Kommune unterschiedlich sein. Dies kann zu signifikanten Unterschieden im Strompreis, bei gleichen Energiebeschaffungskosten, führen.

Beispielweise beläuft sich der Unterschied zwischen der Stadt Hamburg und der Kreisstadt Meißen teils auf bis zu 28% im Strompreis. Für diese Arbeit ist im ersten Zug jedoch nicht relevant von welchem Standort der Strompreis stammt, da sich der konkrete Aufbau in verschiedenen Strompreismodellen unterscheiden kann. Daher wird der Strompreis stets als kumulativer Wert aus Energiebeschaffungskosten und Abgaben angegeben. Ein Beispiel für die Aufteilung des Strompreises zeigt Abbildung 2:

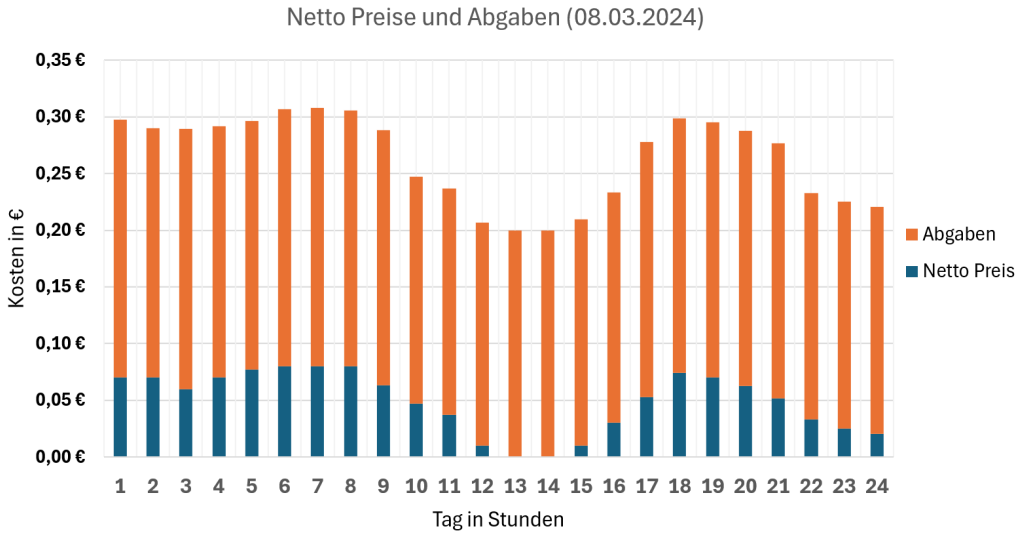

Abbildung 2: Aufteilung des Strompreises am 08.03.2024 (entnommen [Wie24] S.4 Abb. 5 und angepasst)

Anhand der Abbildung lässt sich einer der initialen Motivationsgründe für dynamische Strompreise erkennen: die Kosten der Energiebeschaffung sinken zeitweilig auf und teilweise sogar unter 0,00 cent je kWh. Mittels dynamischer Strompreise kann ein Verbraucher dieses Preisminimum in seinen Verbrauch einplanen, um seine Gesamtstromkosten zu senken; jedoch wirken die im Verhältnis sehr hohen Abgaben diesem Optimierungseffekt entgegen.
Zur optimalen Nutzung des variierenden Strompreises benötigt der Endabnehmer auch eine Stromspeichermöglichkeit mit genug Kapazität, um seinen Verbrauch über die Hochpreisperioden zu decken. Diese verursacht zusätzliche Kosten und wird generell unwirtschaftlich, wenn das Ersparnis aus dynamischen Preisen durch ausgleichende Abgaben gemindert wird. In einem Modell hingegen, in welchem die steuerlichen Abgaben auf sehr niedrige Energiebeschaffungskosten reduziert oder gar nicht erhoben werden, ist die Investition in einen Stromspeicher deutlich lukrativer für den Endverbraucher. Ein solches Modell ist beispielsweise das zuvor erwähnte *DynPlus*, welches in einem folgenden Abschnitt detailliert beschrieben wird.

Ein Tarif für dynamische Strompreise setzt einen sogenannten „intelligenten Stromzähler“ – auch *Smart Meter* – voraus. Dieser zeichnet sich im Gegensatz zu analogen Stromzählern dadurch aus, dass er eine Erfassung und Übertragung des Stromverbrauchs in Echtzeit ermöglicht. Durch die automatisierte Datenübertragung zum Energieversorger ist eine minutengenaue Abrechnung möglich, um somit den Preis je Kilowattstunde zeitabhängig variieren zu können. In vielen Haushalten sind solche Zähler noch nicht verbaut, weswegen sie in dynamischen Strompreistarifen mit angeboten werden. Dies wiederum führt zu einem sogenannten *Lock-In Effekt*, welcher es Kunden erschwert, in Zukunft einen Anbieterwechsel durchzuführen. Der Zeitraum zwischen den Preisschwankungen liegt hierbei in der Regel bei einer Stunde. Üblicherweise werden Preise 24 Stunden zuvor festgelegt und den Kunden zur Verfügung gestellt. Dieser Zeitraum ist typisch für dynamische Strompreistarife, die an Endverbraucher gerichtet sind, wobei an der Strompreisbörse auch kleinere Zeiteinheiten existieren. Preise werden hierbei stets 24 Stunden zuvor öffentlich bekannt gegeben. Der Energieversorger erwirbt den Strom für den Endverbraucher an der Strompreisbörse, wobei teilweise eine Preisaggregation auf die volle Stunde durchgeführt wird. Wie erwähnt sind diese dynamischen Strompreise sowie historische Werte aus verschiedenen Quellen öffentlich verfügbar, beispielsweise über eine entsprechende API-Schnittstelle [Ene25], angeboten vom Fraunhofer-Institut für Solare Energiesysteme ISE [Fra25].

2.2 Themen Relevanz & Abgrenzung

Dynamische Strompreise sind nicht nur auf Grund der erkennbaren wirtschaftlichen Vorteilen (z.B. dass Strom teilweise günstiger bezogen werden kann) sinnvoll, sondern auch aus regulatorischen und umweltrelevanten Gründen. Laut Statistischem Bundesamt stammten im Jahr 2024 ca. 59% des in Deutschland erzeugten Stroms aus erneuerbaren Quellen [Bun25b]. Der Anteil von Photovoltaik allein stieg laut derselben Quelle um 10,4% und bildet damit den zweitgrößten Lieferanten erneuerbarer Energie nach der Windenergie. Daher ist eine Zunahme der täglichen Schwankungen im Strompreis zu erwarten. Gleichzeitig fehlt es momentan noch an Anreizen für Verbraucher, ihr Stromkonsumverhalten anzupassen. Dynamische Strompreise können dazu beitragen die Stromnetze zu entlasten, indem sie einen wirtschaftlichen Anreiz für Endverbraucher darstellen ihr Konsumverhalten, wenn möglich, anzupassen. Gleichzeitig bietet dies einen weiteren Anreiz für den Ausbau von Photovoltaik und anderen erneuerbaren Stromquellen, da sich hier ein noch unerschlossener Markt auftun könnte.

Dafür benötigt es jedoch Strompreismodelle, die den zeitlichen Preisvorteil für den Endverbraucher realisierbar machen. Im Zuge dieser Arbeit werden die rechtlichen und operativen Herausforderungen für die Einführung solcher Strompreismodelle aber nur rudimentär betrachtet und nicht näher untersucht. Ein weiterer Faktor, den es zu beachten gilt, ist der Ausbau persönlicher Stromspeicher, die für die effiziente Ausnutzung eines dynamischen Preises nötig wären. Da das extreme Niedrigpreissegment meist nur die Mittagsstunden umfasst, benötigen Endverbraucher einen Speicher, um während der Hochpreisperioden unabhängig vom Netz zu sein. Somit können dynamische Strompreise auch einen Anreiz für den privaten Ausbau von Stromspeichern bieten, was eine zusätzliche Verteilung des Konsumverhaltens nach sich ziehen würde. Da die Belastung des Stromnetzes zukünftig eher steigen als abnehmen wird, wäre ein Abschwächen der Verbrauchsspitzen von Vorteil. Eine gleichmäßigere Verteilung des Stromverbrauchs würde auch dazu führen, dass eine geringe Produktionskapazität notwendig wäre, um den Bedarf zu decken.
Diese Vorteile lassen sich - in der Theorie - aus einem dynamischen Strompreis gewinnen aber die tatsächliche Auswirkung in der Realität bedarf einer separaten Untersuchung, welche den Rahmen dieser Arbeit übersteigen würde. Die hier beschriebene Simulation und Forschung hat einen explorativen Charakter um mögliche Alternativen zum aktuellen Strompreismodell zu evaluieren. Für eine Einführung dieser Modelle wären jedoch Tests in realen Umgebungen unabdinglich, welche zum jetzigen Zeitpunkt noch nicht durchgeführt werden können.

2.3 Diskrete Simulation

Simulation ist ein flexibles Analysemittel, welches in vielen Teilen der Wissenschaft Anwendung findet. Dabei wird ein Modell erstellt, welches versucht einen bestimmten Vorgang darzustellen, wobei die Realität vereinfacht wird, jedoch wesentliche Aspekte beibehalten werden. Simulationen können mittels Rechentechnik automatisiert werden und beinhalten oft zufällige oder variable Komponenten vgl. [MM89] S.1f. Da die Zeit eine essentielle Komponente in den meisten Simulationsmodellen darstellt, kann ihre Modellierungsform als Klassifikationsmerkmal verwendet werden. Abbildung 3 zeigt eine solche Unterteilung von Modellen nach Art der Zeit.

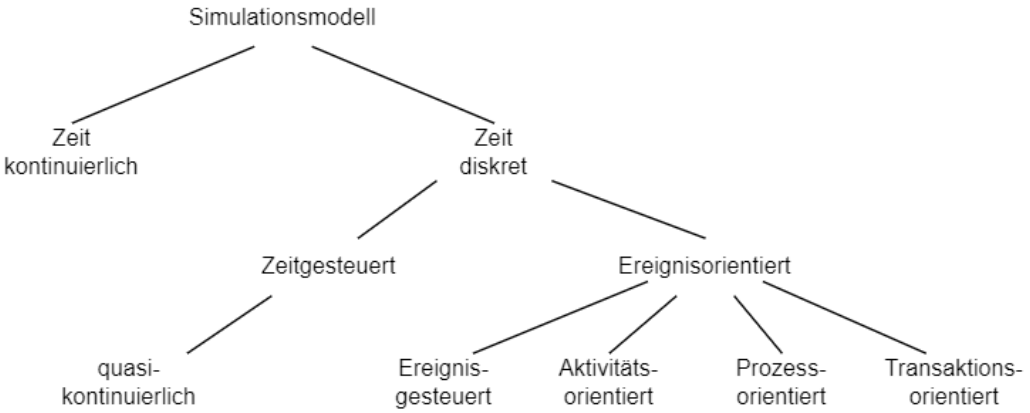

Abbildung 3: Klassifikation von Simulationsmodellen, nach [MM89] Abb. 1

Der Themenbereich dieser Arbeit bezieht sich konkret auf diskrete Simulation, welche sich durch eine 'plötzliche' oder 'sprunghafte' Wertänderung kennzeichnet. Am Beispiel der Vorhersage dynamischer Strompreise stellt sich dies durch den Wechsel des Strompreises innerhalb der festen Zeitabschnitte da, welche im Modell definiert sind. Da es bei der Simulation von Strompreismodellen nach einer gewissen Anzahl Zeitschritte immer zu Wertänderungen kommt, handelt es sich auch stets um diskrete Simulationen. Da der Wertwechsel dabei zeitgesteuert ist und kein festes Ende oder Abbruchkriterium existiert, außer den Zeitrahmen welcher Simuliert wird, kann man bei der dynPreis Simulation auch von einer quasi-kontinuierlichen Simulation sprechen.

Der Einsatz von Simulationen kann unterschiedlichen Zielen dienen, wobei man einige übergreifende Ziele erkennen kann, welche sich auch auf den Kontext dieser Arbeit beziehen lassen, nach [Pag+88]:

Optimierung: Die Suche nach einer optimalen Belegung der Variablen, welche Engpässe vermeidet oder Werte minimiert/maximiert. Im Kontext von dynamischen Strompreisen kann dies beispielsweise eine Maximierung der Ersparnisse gegenüber Fixpreisen bedeuten.

Entscheidungsgrundlage: Vergleichen von verschiedenen alternativen Modellen. Hierbei handelt es sich um eines der Primärziele, welches mit dieser Arbeit verfolgt wird.

Prognose: Vorhersage der Entwicklung des Systemverhaltens. Diese Aufgabe wird von den hier betrachteten Modellen nur indirekt erfüllt, zwar findet eine Prognose des Preises statt, jedoch werden keine Änderungen am Systemverhalten selbst betrachtet.

Überprüfung von Theorien: Grundlegender Baustein alternativer Strompreismodelle - alle zielen darauf ab, eine Theorie über mögliche Preismodelle zu überprüfen

Validierung: Korrektes Verhalten des Systems. Die Validierung, welche im Zuge dieser Arbeit betrachtet wird, bezieht sich auf die korrekte Implementierung und Arbeitsweise des Systems und ist nicht das Ziel der Simulation.

Modellanimation: Darstellung von Systemverhalten, spielt eine untergeordnete Rolle, wird aber in Zusammenarbeit mit der einleitend erwähnten dynamischen Darstellung relevant.

2.4 Eingesetzte technische Lösungen

Für den Vergleich verschiedener Simulationsmodelle und zur Auswertung der daraus entstehenden Informationen werden, verschiedene technische Mittel benötigt. In diesem Abschnitt werden die eingesetzten Softwareprodukte und Programmiersprachen kurz vorgestellt und deren Verwendung begründet. Die genaue Umsetzung dieser Lösungen wird im Kapitel 3 (Planung) und im Kapitel 4 (Entwicklung) genauer betrachtet.

Zum Vergleich verschiedener Modelle und deren Simulationsergebnisse wird eine Datenbank benötigt, welche die Informationen speichert, aggregiert und bereitstellt. Ebenso benötigt das Data Warehouse als Datengrundlage für die Business Intelligence-Funktionen eine spezielle Datenbank. Beide Anforderungen wurden mit demselben Softwareprodukt, dem SQL Server 2022 der Firma Microsoft, abgedeckt. Der SQL Server bietet die Möglichkeit, verschiedene relationale Datenbanken gleichzeitig bereitzustellen und wird mittels der Datenbanksprache SQL (Structured Query Language) verwaltet. Zudem bietet dieses Produkt die Option, um viele Funktionalitäten erweitert zu werden, beispielsweise den SQL Analytics Services, welche verwendet werden können, um ein Data Warehouse aufzubauen. Damit bildet der SQL Server 2022 alle nötigen Datenbankfunktionen für die Bearbeitung der Forschungsfrage in einem Softwareprodukt ab. An dieser Stelle muss erwähnt werden, dass der SQL Server in verschiedenen Lizenzierungsoptionen mit unterschiedlichen Funktionalitäten angeboten wird, namentlich Express, Developer, Standard und Enterprise. Express ist eine kostenfreie, stark eingeschränkte Version des SQL Servers, hauptsächlich vorgesehen für private Anwendungen und kleine Teststellungen. In dieser Version kann eine Datenbank maximal 10 Gigabyte groß sein und es wird nur bis zu 1 GB RAM für die Operation der Software verwendet. Außerdem sind verschiedene Funktionen wie die Volltextindizierung oder der SQL Server-Agent nicht verfügbar. Daher war die Express-Version, trotz ihrer Kostenvorteile, für die Zwecke dieser Arbeit nicht geeignet. Stattdessen wurde die Developer-Lizenz eingesetzt, welche den vollen Funktionsumfang des SQL Servers ermöglicht, aber nur für Entwicklungs- und Forschungsarbeiten verwendet werden darf. Sollten zu einem späteren Zeitpunkt die für diese Arbeit entwickelten Werkzeuge produktiv oder mit Gewinnabsichten verwendet werden, muss entweder eine Standard- oder eine Enterprise-Lizenz eingesetzt werden. Dabei entspricht nur die Enterprise-Lizenz dem gleichen Funktionsumfang wie die Developer-Lizenz [Cor22a]. Eine Aufschlüsselung der Lizenzen wird in Abbildung 4 dargestellt.

SQL Server 2022 Edition	Licensing Options	
	Server + CAL	Per Core
Enterprise		•
Standard	•	•
Developer	Free edition	
Express	Free edition	

Abbildung 4: Übersicht der SQL Server Lizenzierung entnommen dem SQL Server Lizenz Datenblatt

Sollte ein solcher Lizenz wechsel vollzogen werden, muss beim Einsatz einer Standard-Lizenz noch auf die Lizenzierung der sogenannten CALs geachtet werden. Hierbei handelt es sich um die Zugriffslizenz für jeden Agenten, welcher die Funktionen des SQL Servers in Anspruch nimmt.
Beim Einsatz einer Enterprise-Lizenz entfällt diese Notwendigkeit [Cor22a] S. 14 f. Der SQL Server wurde in der anfänglichen Testphase zunächst auf einem leistungsstarken Windows 11-Notebook betrieben. Mit voranschreiten des Projekts war dies jedoch nicht mehr zweckdienlich, weswegen der SQL-Server auf einen Windows Server 2022 Datacenter migriert wurde. Dies war nicht nur aufgrund der besseren Verwaltbarkeit und Skalierbarkeit nötig, sondern auch, um die initial erwähnte Darstellung der Daten auf einer dynamischen Website zu ermöglichen. Damit die Daten abgerufen werden können, müssen sie jederzeit bereitstehen, was auf einem Endnutzergerät nicht gegeben war.
Für die Entwicklung der verschiedenen Skripte in Python, Transact-SQL und MDX wurde Visual Studio Code, ebenfalls von der Firma Microsoft, verwendet. Für die Konfiguration und Abfrage des geplanten Data-Warehouse-Cubes wurde Visual Studio 2022 mit der Erweiterung *Microsoft Analysis Services Projects 2022* [Cor22b] eingesetzt. Mit diesen Werkzeugen wurden auch die BI-Funktionalitäten realisiert, wobei die Darstellung mittels Microsoft Excel erfolgt. Alternativ ließe sich die Darstellung auch mittels Microsoft Power BI umsetzen, um eine Integration in die Azure-Cloud zu ermöglichen. Dies würde zugleich eine leichtere Einbindung in cloudbasierte Geschäftsprozesse erlauben.
Eine vollständige Abbildung aller entwickelten Softwarelösungen in der Azure-Cloud wurde zu Beginn der Arbeit in Betracht gezogen. Aufgrund der damit verbundenen hohen Kosten konnte dieser Ansatz jedoch nicht gerechtfertigt werden, weshalb stattdessen kostengünstigere virtuelle Maschinen des Anbieters Strato verwendet wurden.

Fullstack-Cloudanbieter wie Microsoft Azure oder Amazon AWS stellen das vollständige Produktportfolio einer modernen Unternehmens-IT-Landschaft bereit und könnten im Falle eines produktiven Einsatzes der im Rahmen dieser Arbeit entwickelten Lösungen in Erwägung gezogen werden. Für diese Arbeit wurden zwei virtuelle Maschinen eingesetzt und konfiguriert: die Maschine für den zuvor erwähnten Windows Server und eine als Webserver mit Ubuntu 22.04 LTS.
Die Arbeit selbst wurde in LaTeXmittels TexStudio geschrieben, für die Quellenverwaltung wurde Mendely eingesetzt und die Darstellung im Dokument erfolgt mittels BibTex.

2.5 Multimodel Validierung

Bevor der Multimodel-Aspekt beschrieben werden kann, muss zunächst der Begriff *Validierung* im Kontext dieser Arbeit definiert werden. Unter Validierung von Simulationsmodellen versteht man die Überprüfung der korrekten Abarbeitung der vorgegebenen Prozessschritte, sowie die Nachvollziehbarkeit der prognostizierten Ergebnisse. Zusätzlich müssen sich diese Ergebnisse in einem realistischen Rahmen bewegen [Mes+17]. Für diese Arbeit bedeutet dies insbesondere die Überprüfung verschiedener Implementierungen einzelner Preismodelle gegeneinander.
Multimodel in im Kontext dieser Arbeit wiederum bedeutet, dass verschiedene Modelle mit der gleichen Zielstellung gegen dieselben Parameter validiert werden. Dabei spielt nicht nur der Grad der Zielerreichung eine Rolle, sondern auch der direkte Vergleich der Modelle untereinander. Für den parallelen Vergleich verschiedener, gleichgerichteter Modelle existieren unterschiedliche Ansätze [MTK18]. Im Zuge dieser Arbeit wurden einige ausgewählte Ansätze betrachtet, wobei festgehalten werden muss, dass kein allgemeingültiges, sondern viele verschiedene anwendungsspezifische Modelle existieren. Hierbei war die Arbeit von [AF17] eine sehr gute Basis, welche nicht nur den aktuellen Stand der Wissenschaft im Bereich der Multi-Modell-Vergleiche zusammenfasst, sondern auch einen spezifischen Ansatz das *Multimodel Framework* (im Folgenden auch MMF) detailliert beschreibt. Dabei handelt es sich um ein theoretisches Modell, welches sich sehr gut eignet, um auf den Bereich der dynamischen Strompreis-Validierung angewendet zu werden. Das MMF ist ein Ansatz, welcher einen Fachbereich in verschiedene Modelle (i.e. Simulationsmodelle) zerlegt und einen gewichteten Gesamt-Output liefert. Hierbei wird der zu simulierende Wertebereich auch als *Operation Space* bezeichnet, welcher in *Operating Regions* aufgeteilt wird. Im Multimodel-Framework werden diese Regionen mittels Submodellen, welche auch als lokale Modelle bezeichnet werden, abgebildet.

Dabei handelt es sich um verschiedene Simulationsmodelle mit derselben Zielstellung; in Bezug auf die Strompreisfindung ist der *Operation Space* alle möglichen Strompreise und Szenarien, während die *Operating Regions* einzelne Preismodelle darstellen [AF17], S.1150 ff. Die folgende Grafik zeigt, wie der Operation Space mittels verschiedener Modelle abgedeckt wird:

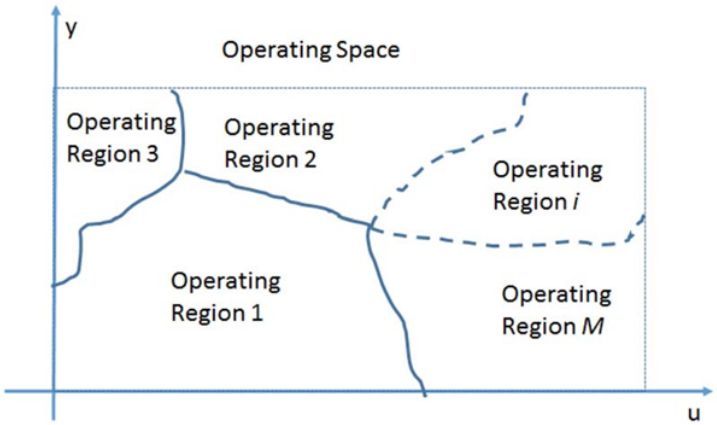

Abbildung 5: Zerlegung des Operation Space nach MMF, entnommen [AF17] S.1150 Fig.2

Hierbei lassen sich diese Modelle im MMF nach vier Attributen kategorisieren [AF17], S.1151 f. :

Partition Strategy: Beschreibt die Trennung der lokalen Modelle nach Ausrichtung im Operation Space, auf welche Werte optimiert wird, bzw. möglichst konstante Preise zu halten oder das häufige Erreichen von minimalen Preisen.

Submodel Structural Identification: Aufteilung der Modelle nach interner Strukturierung, lineare oder nichtlineare Arbeitsweise.

Transition Between Models: Vernetzung der Modelle untereinander, beschreibt in welchem Ausmaß und mit welcher Auswirkung Werte, aus einem Modell, in ein weiteres im selben Operation Space fließen.

Method of Realization: Wie die Modelle zusammen agieren, um den ganzen Operation Space abzudecken. Wie die Gesamtheit des Werteraums abgebildet wird.

Die Ergebnismengen der einzelnen Modelle müssen wiederum in einer gewichteten Zielfunktion zusammengefügt werden. Dabei wird zwischen *hard switching* und *soft switching* unterschieden. Bei ersterer Variante wird entweder das gesamte Ergbnisset in die Zielfunktion auf genommen oder kein Teil der Ergebnisse. Bei *soft switching* findet eine Abstufung statt, was bedeutet das Ergebnismengen auch nur teilweise einfließen können. Die folgende Abbildung verdeutlicht den Prozess:

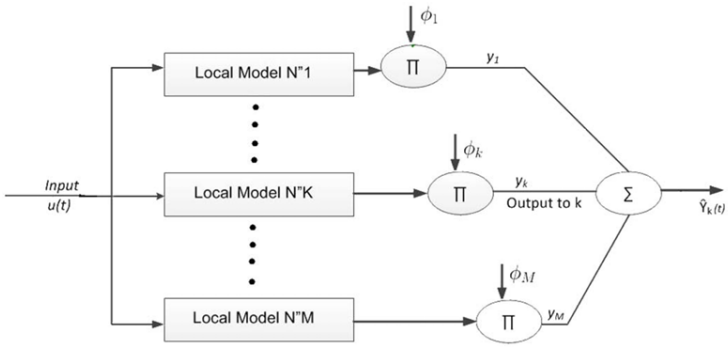

Abbildung 6: Zusammenführen lokaler Modelle, entnommen [AF17] S.1151 Fig.3

Während MMF eine sehr gute theoretische Basis bietet, beschreiben Yilin Huang und Igor Nikolic eine praxisorientierte Lösung, um unabhängige Modelle parallel zu steuern und zu verwalten [HN24]. Mit der Betrachtung der praktischen Umsetzung wird auch ein generelles Problem bei der Validierung der Simulationsergebnisse für alternative Strompreismodelle klar. Wie eingangs beschrieben, bezeichnet Validierung unter anderem eine Überprüfung der Daten auf Korrektheit, was in der Praxis üblicherweise gegen reale Messdaten geschieht. Diese existieren bei alternativen Strompreisdaten jedoch nicht; zwar kann gegen den tatsächlichen Strompreis verglichen werden, dies bietet aber keine Rückschlüsse auf die Korrektheit des Modells. Eine mehrfache unabhängige Implementation des gleichen Algorithmus auf einer einheitlichen theoretischen Beschreibung erlaubt es, zumindest die Preismodelle in sich zu validieren. Dafür muss jedoch eine Implementation eines bestimmten Algorithmus als korrekt angenommen werden, gegen welche die anderen validiert werden. Durch die Rotation des *korrekten* Algorithmus kann sichergestellt werden, dass alle Implementationen das zugrundeliegende Modell korrekt umsetzen. Zu diesem Zweck kann jeder simulierte Wert, einer bestimmten Modell-Implementation mit jedem anderen verglichen werden und bei gleichen Ausgangsparametern, darf es nur zu minimalen Abweichungen kommen. Abweichungen können durch unterschiedliches Runden bei verschiedenen Programmiersprachen entstehen. Durch diese Vorgehensweise lassen sich Fehler in der Modellimplementierung schnell identifizieren, da in der Ergebnismenge ein entsprechendes Muster bei falschen Berechnungen entsteht.
Es wurde jedoch nicht nur der Ansatz des Multimodel Framework betrachtet, sondern auch weitere Multi-Modell-Validierungsvorgehensweisen wie die in [Aja+06] vorgestellte Methode. Die Methode wirkte vielversprechend, konnte aufgrund der Notwendigkeit realer Daten als Validierungsgrundlage jedoch nicht für diese Arbeit umgesetzt werden. Dennoch konnte zusätzliches Verständnis für unterschiedliche Validierungsansätze gewonnen werden.

2.6 Bestehendes DynPlus Preismodell

Wie bereits initial erwähnt, war einer der Anstöße für diese Arbeit das von Prof. Wiedemann vorgestellte dynamische Strompreismodell *DynPlus*. Da es sich dabei um das erste Modell handelt, welches validiert werden sollte und zum besseren Verständnis der Thematik alternativer Preismodelle, wird es im Detail erklärt. Erstmalig vorgestellt wurde das Modell in [Wie24] und weitere Ergebnisse wurden im Rahmen der *Annual Modeling and Simulation Conference (ANNSIM) 2024 und 2025* präsentiert. Der Ansatz des Modells ist eine Begrenzung der steuerlichen Abgaben auf ein Maximum von 200%, um das Nutzen von sehr niedrigen bis negativen Preisen zu ermöglichen, während die fehlenden Abgaben auf das Mittelpreissegment verlagert werden. Dies wird zusammengefasst in der gegebenen Legislatur-Empfehlung: *„Die Abgabenlast bei dynamischen Strompreisen darf 200% des Netto-Strompreises nicht übersteigen. Die summarische Differenz zur bisherigen Berechnung dynamischer Brutto Strompreise kann zu anderen Zeiten durch Aufschläge ausgeglichen werden.“* [Wie24], S. 6. Die durch dieses Modell entstehende Preisverschiebung, lässt sich in Abbildung 7 erkennen; die realen Preise für den Tag sind in Grau darübergelegt.

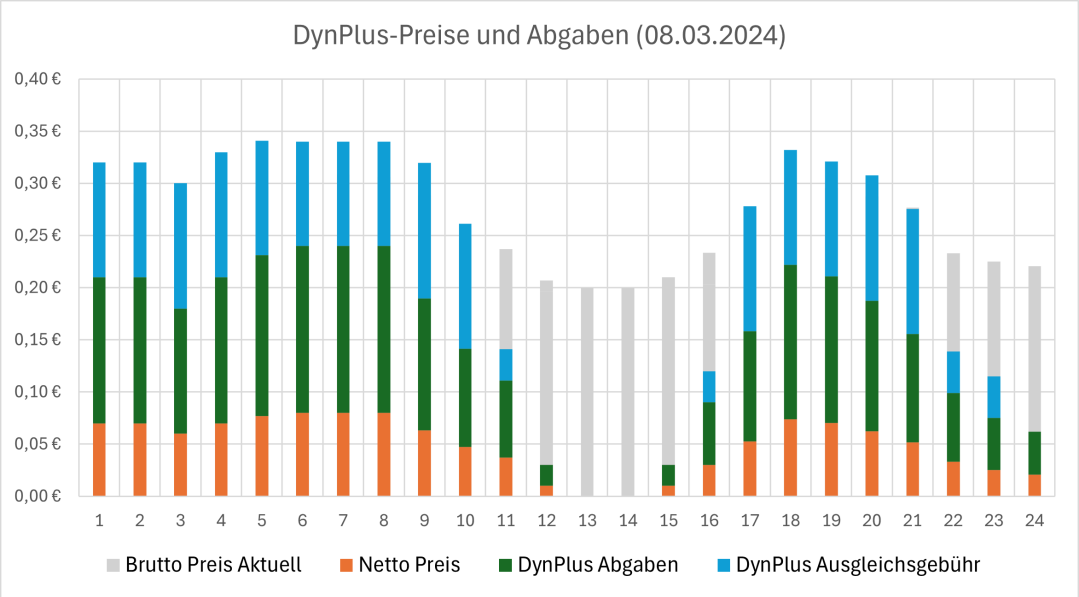

Abbildung 7: Beispiel für die Preisverschiebung

Durch die prozentuale Berechnungsweise fällt der Strompreis im *DynPlus*-Modell bei Energiebeschaffungskosten von 0,00 Cent/kWh auf ebenfalls 0,00 Cent/kWh. Um den dadurch entstehenden Steuerverlust auszugleichen, wird ein Abgabenaufschlag auf die Mittelniveau-Preise erhoben. Das größte Potenzial hat ein Verbraucher in diesem Modell, wenn er einen privaten Stromspeicher besitzt, da er diesen dann zu den Null- oder Niedrigpreiszeiten aufladen und zu Hochpreiszeiten verbrauchen kann. Daraus ergeben sich verschiedene Simulationsszenarien, beispielsweise ob ein Speicher eingesetzt wird, welche Kapazität er hat oder ob Photovoltaik im System vorhanden ist. Nicht alle diese Szenarien wurden abschließend untersucht.
Zusätzlich zur Auswirkung auf die Verteilung der Tagespreise wurden auch die Langzeitauswirkungen von *DynPlus* auf den Ausbau von privaten Stromspeichern simuliert. Unter der Voraussetzung, dass ein Endverbraucher stets wirtschaftlich handelt, lässt sich annehmen, dass dieser einen Speicher für den privaten Gebrauch erwerben wird, solange er diesen mittels Kostenersparnissen refinanzieren kann und sein persönlicher Speicherbedarf noch nicht gedeckt ist. Die Kostenersparnisse entstehen in diesem Fall durch die Ausnutzung der Niedrigpreise im *DynPlus*-Modell und das Abfedern der Hochpreise durch Nutzung des Speichers. Die Ergebnisse dieser Simulation sagen einen Anstieg der privaten Stromspeicherkapazitäten um bis zu 1366% zum Jahr 20250 voraus [Wie24] S. 7.

3 Planung

3.1 Systementwurf und Modellierung

Mit den grundlegenden theoretischen Konzepten aufgearbeitet, ist es nun notwendig eine Gesamtstruktur zu entwickeln, welche die Forschungsfrage potenziell umsetzten kann. Dabei wird auf zwei unterschiedliche Datenbanken gesetzt: eine Online Transaction Processing Datenbank für die Kommunikation mit den Simulationsmodellen und eine Data Warehouse Datenbank für das Auswerten und Vergleichen der verschiedenen Preismodelle. Basierend auf der DWH-Datenbank wird zudem der Cube (OLAP-Würfel) generiert, welcher für die geplanten Business Intelligence Funktionen notwendig ist. Abbildung 8 zeigt eine Darstellung der geplanten Struktur.

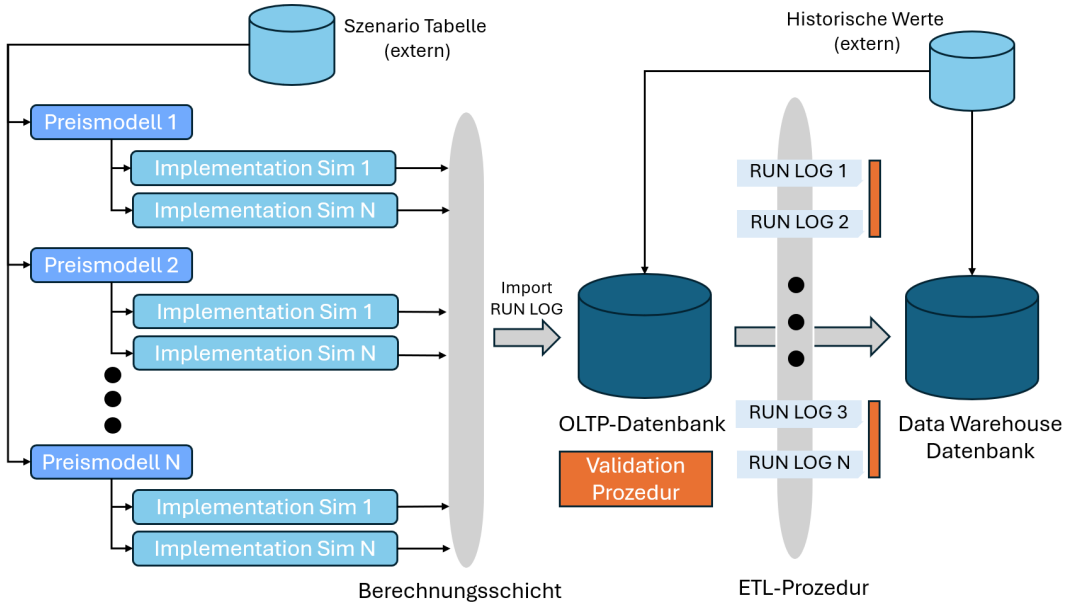

Abbildung 8: Darstellung der geplanten Gesamtstruktur

Wie zuvor erwähnt, ist eines der Ziele die Validierung von verschiedenen Implementationen der unterschiedlichen Preismodelle, wobei diese Aufgabe primär in der OLTP Datenbank durchgeführt wird. Für Auswertungszwecke werden diese Ergebnisse auch in die Data Warehouse Datenbank überspielt. Dargestellt ist dies durch die Unterteilung der Preismodelle in einzelne Implementationen. Im Schritt der Validierung soll die korrekte Abarbeitung der einzelnen Implementationen sichergestellt, im Data Warehouse hingegen die verschiedenen Preismodelle miteinander und gegen historische Werte verglichen werden. Dabei ist anzumerken, dass historische Werte aus einer externen Quelle bezogen werden, hierbei kann es sich um Werte von der Strompreisbörse oder den Datenbeständen eines Energiedienstleisters handeln.

Ebenso werden die Szenarien, welche die Preismodelle simulieren sollen, extern bereitgestellt. Die tatsächliche Durchführung der einzelnen Simulationen ist nicht der Fokus dieser Arbeit und wird daher in der Berechnungsschicht zusammengefasst. Wo die Simulation durchgeführt wird, in der Cloud oder auf einer externen Rechenressource, hat dabei keine direkte Relevanz. Der Fokus dieser Arbeit und damit auch dieses Aufbaus liegt ganz auf der Speicherung, Aggregierung und Auswertung der simulierten Werte. Die Simulationsergebnisse (RUN LOG) werden nach der Generierung zuerst in die OLTP-Datenbank geladen, dies geschieht beispielsweise über einen CSV Importierungsjob mittels Stored Procdures. Es ist vorgesehen wenige verschiedene Formate anzubieten, in denen Daten für die Auswertung eingereicht werden können. Es sollen zwar unterschiedlich strukturierte Modelle ausgewertet werden, dennoch ist eine gewisse Vereinheitlichung der Eingangsdaten nötig. Die vollständigen Simulationsläufe sollen jedoch nicht dauerhaft in der OLTP Struktur gespeichert werden, diese sind für die Validierung einer Implementation nur einmalig nötig, daher werden sie nach kurzer Zeit automatisch in die DWH-Datenbank verschoben. Dabei werden in der OLTP-Datenbank nur zusammenfassende Monatswerte behalten, um einen schnellen allgemeinen Vergleich verschiedener Modelle zur ermöglichen. Die OLTP-DB ist für einen operativen Betrieb vorgesehen, also das häufige hinzufügen und abrufen von Datensätzen. Innerhalb der Data Warehouse Datenbank werden die vollständigen Simulationsläufe benötigt, um daraus Kennzahlen (Key Performance Indicator - KPI) generieren zu können und um komplexe Analysen der Daten zu ermöglichen. Der daraus resultierende Aufbau wird auch als Relational Online Analytical Processing (ROLAP) bezeichnet. Dieser kennzeichnet sich dadurch, dass der Multidiminsonale Cube, welcher aus der Data Warehouse Datenbank generiert wird, keine statische Struktur hat, sondern dynamisch auf den SQL Abfragen des Nutzers erstellt wird [GC06] S.164 ff. Dies wird einem statischen Cube vorgezogen, zwar bietet dieser performantere Abfragen als die dynamische Variante, aber Anpassbarkeit an die sich wechselnden Anforderungen innerhalb eines laufenden Projekts sind in diesem Fall von größerem Nutzen. Die für verschiedene Vergleiche notwendigen historischen Daten sollen automatisiert von einer externen Schnittstelle abgerufen werden. Hierbei kann es sich um Werte von der Strompreisbörse oder den Datenbeständen eines Energiedienstleisters handeln.

3.2 Schnittstellendefinition

Da es Vorgesehen ist, dass die zuvor beschriebene Gesamtstruktur von einer Vielzahl an unterschiedlichen Akteuren verwendet werden soll, wurde es als sinnvoll erachtet die Schnittstellen des Systems nach außen zu identifizieren und beschreiben. Zu diesem Zweck wurde eine Schnittstellendefinition, angelehnt an ein UML (Unified Modeling Language) [SDO17] Komponentendiagramm erstellt, womit Zugriffe besser erkenntlich werden. Abbildung 9 zeigt die Darstellung:

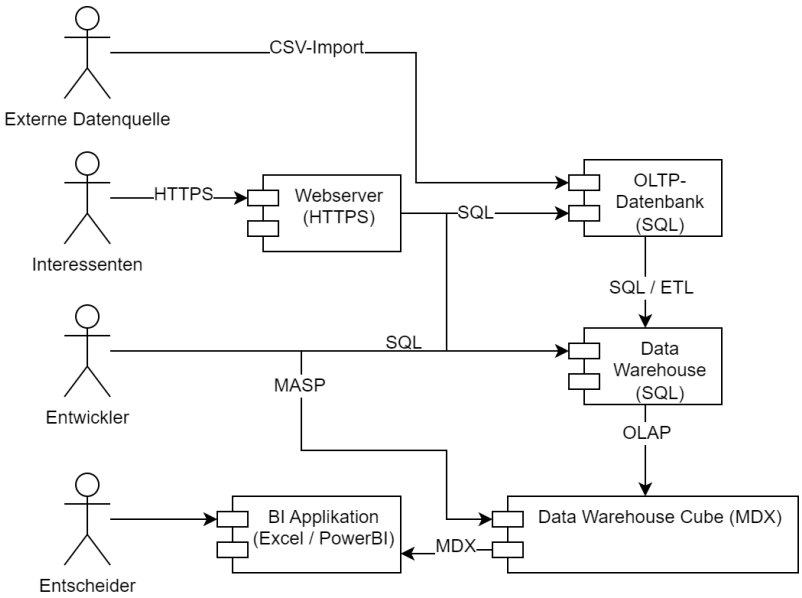

Abbildung 9: Komponentendiagramm der Gesamtstruktur

Verschiedene Komponenten haben Schnittstellen nach außen: die OLTP-Datenbank, die Data Warehouse Datenbank, der darauf aufbauende Data Warehouse Cube, der Webserver sowie die verwendeten BI-Applikationen. Diese Komponenten interagieren über standardisierte Protokolle wie SQL, MDX, HTTPS und CSV-Import.
Externe Akteure, die mit dem System interagieren, sind:

Interessenten: rufen über den Webserver allgemeine Auswertungen und Zusammenfassungen ab

Entscheider: führen Analysen mittels BI Applikationen auf dem Data Warehouse Cube aus

Externe Datenquellen: stellen historische Strompreisdaten oder simulierte Werte in CSV Dateien bereit

Entwickler: kommunizieren mit allen Komponenten, über deren entsprechende Schnittstellen

Diese Struktur ermöglicht eine klare Trennung zwischen operativer Datenverarbeitung und analytischer Auswertung, während gleichzeitig externe Datenquellen flexibel angebunden werden können. Die Beschreibung der Schnittstellen erfolgt hierbei relativ allgemein, da es sich zum einen, bei der Kommunikation mit den Datenbanken um SQL - also eine standardisierte Sprache - handelt und zum anderen da die für die CSV Imports verwendeten Skripts sich je nach expliziten Anwendungsfall sehr leicht anpassen lassen. Es ist weder davon auszugehen, dass Fachfremde Anpassungen an den Schnittstellen vornehmen müssen, noch das viele neue Akteure gleichzeitig auf das System zugreifen. Ausnahme sind hierbei externe Akteure der *Interessenten*-Gruppe, deren Schnittstelle jedoch darauf ausgelegt ist.

3.3 Datenstrukturen & ERM

Wie bereits bei der Beschreibung des Systementwurfs in Abbildung 8 erwähnt, werden zwei unterschiedliche Datenbanken benötigt. Die erste davon, welche auch für die Ausführung der Validierung verwendet wird ist eine Online Transaction Processing-Datenbank. Zusätzlich liegt die Aufgabe dieser Datenbank in der Vereinheitlichung der unterschiedlichen Preismodelle und deren Implementationen, in eine Datenstruktur, ohne das dabei ein Informationsverlust entsteht. Dies folgende Abbildung zeigt das zugehörige Entity-Relationship-Modell Diagramm.

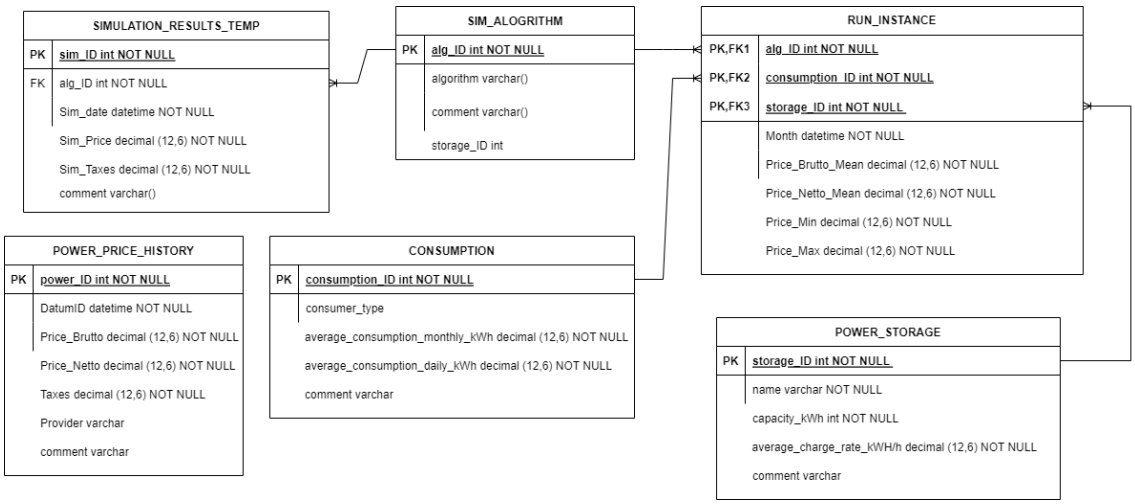

Abbildung 10: ERM der OLTP Datenbank

Die Simulationsergebnisse werden zunächst in die Tabelle *SIMULATION_RESULTS_TEMP* geladen, wobei Preismodell und Implementation gleichermaßen mittels der Algorithmus ID unterschieden werden. Hierfür wurde festgelegt, dass unterschiedliche Preismodelle durch die Hunderterstelle gekennzeichnet werden, während Implementation durch die Zehner und Szenarien durch die Einerstelle unterschieden werden. So ist beispielsweise der Algorithmus mit der ID 107 eine Implementation des *DynPlus* Modells (**1**07) dabei handelt es sich um die erste Implementierung (1**0**7), des 7 Szenarios (10**7**). Das Kommentarfeld in der Tabelle *SIM_ALGORITHM* enthält einen Kompositschlüssel, dieser stellt in einer menschlich lesbaren Form alle Informationen zu einer bestimmten Algorithmus-Ausführung dar. Zwar entsteht dadurch eine leichte Informationsdopplung, da Teile der Inhalte in anderen Tabellen abgebildet werden, aber dies wird akzeptiert um eine bessere Lesbarkeit für Anwender zu erreichen. Ein Beispiel für einen solchen Schlüssel ist: **DYNPLUS_STDPRICE035_DynUse005_Power1KW_STOR012_-AUFS006_AUFB001_RATIO34_C#NETV80**. Aufgeschlüsselt bedeutet dies; Es handelt sich bei diesem Algorithmus um eine Implementation des *DynPlus* Modells, gefolgt von einigen speziellen Variablen zugehörig zum jeweiligen Modell, zum Beispiel, dass Werte ab 0,05€ als günstig angesehen werden und das ein durchschnittlicher Verbrauch von 1 kw/h besteht. Der letzte Wert weist die entsprechende Implementation aus, in diesem Fall eine Implementierung in C# auf Basis von *.net 8*. Ein solcher Schlüssel ließe sich auch erreichen, indem die Information in separaten Spalten gespeichert und mittels SQL-Anweisung verkettet ausgegeben werden. Des Weiteren existiert eine Tabelle *POWER_PRICE_HISTORY*, in welcher die historischen Strompreise abgelegt werden. Dies ist notwendig um die alternativen Preismodelle im Vergleich mit den Realpreisen darzustellen. Die vollständigen Simulationsläufe (SIM_LOGS) werden nur temporär in der Validerungsstruktur gespeichert, bevor sie mittels des Extract-Transform-Load-Prozesses (ETL) in das Data Warehouse geladen werden. Daher werden, um einen allgemeinen Überblick über bereits durchgeführte Simulationen zu erhalten, aggregierte Werte in der Tabelle *RUN_INSTANCE* gespeichert. Hierbei wird auf eine bestimmte Implementation und Ausführung verwiesen, wobei nur die monatlichen Durchschnittspreise gespeichert werden. Dadurch kann für jedes Modell ein schneller allgemeiner Überblick von verschiedenen Modellen bereitgestellt und kürzlich durchgeführte Simulationen detailliert untersucht werden.

Für den Entwurf der Datenbankstruktur, welche dem Data Warehouse zugrunde liegt, wurde sich für einen Aufbau nach dem Stern-Schema entschieden. Dabei werden die einzelnen Dimensionen des Data Warehouse Cubes um eine zentrale Faktentabelle angeordnet, welche Fremdschlüssel aus jeder Dimension beinhaltet. Da die Daten zuerst in der OLTP Struktur vorbereitet werden, muss das Datenmodell für das Data Warehouse Teile der Struktur widerspiegeln. Abbildung 11 zeigt den Aufbau als ERM-Diagramm: Wie zuvor erwähnt sind alle Tabellen neben der

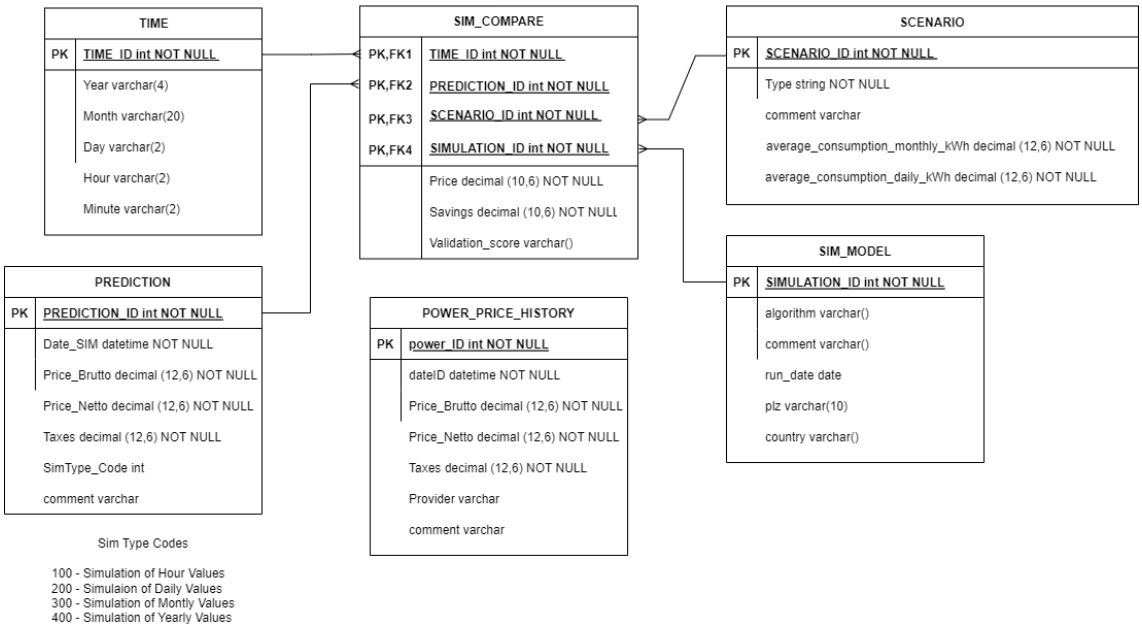

Abbildung 11: ERM der DWH Datenbank

Faktentabelle *SIM_COMPARE* ebenfalls Teil des Cubes als Dimensionstabellen. Eine, die nochmals besonders hervorgehoben werden sollte, ist die Dimension *TIME* (dt. Zeit). Sie dient ausschließlich dazu, den jeweiligen Zeitstempel für einen Datensatz abzuspeichern. In einer normalisierten Datenbank wäre diese Speicherart nicht möglich, auch weil sie sehr ineffizient ist, aufgrund der Notwendigkeit für jede Datumsabfrage einen JOIN durchführen zu müssen. Für ein Data Warehouse ist diese Art der losgelösten Zeit jedoch sinnvoll, da die ineffizienz der JOINs ausgeglichen wird. Dieser Aufbau erlaubt es auch, dass Modelle, welche unterschiedliche Zeitinkremente verwenden, miteinander verglichen werden können. Dies ist eine der grundlegenden Anforderungen an das System, da die Aufteilung in Stunden (wie beispielweise im *Dyn-Plus* Modell) keine feste Vorgabe ist. Somit ist es auch denkbar, dass ein alternatives Strompreismodell mit dreißig Minuten oder fünfzehn Minuten Inkrementen arbeitet.
In beiden Modellen ist der Einsatz von *DECIMAL* als Datentyp für Stromerzeugungskosten, sowie Stromkosten zu beachten.

Ebenso wie *NUMERICAL* bietet dieser Datentyp eine feste Genauigkeit und Anzahl an Dezimalstellen. Dadurch wird sichergestellt, das bei Gleitkommaberechnungen keine Wertfehler auftreten. Diese können bei Einsatz von beispielsweise float als Datentyp auftreten, da sich nicht alle Nachkommastellen eindeutig nicht-periodisch als Binärzahl abbilden lassen.

3.4 Dynamische Datendarstellung

Wie bereits in der Einleitung erwähnt, war es von Interesse die Daten und daraus ableitbaren Erkenntnisse für verschiedene Entscheidungsträger verfügbar zu machen. Dabei lag der Fokus auf der Auswertung mittels BI Mitteln. Diese können jedoch für fachfremde Interessenten zu komplex sein und eignen sich daher nur bedingt für exploratives Verstehen der Thematik. Aus diesen Gründen wurde eine visuelle Darstellung mittels einer dynamischen Website in Betracht gezogen. Die Entwicklung einer solchen Website hätte den Rahmen dieser Arbeit jedoch gesprengt, da die dabei auftretende thematische Komplexität, eine eigene Wissenschaftliche Arbeit erfordern würde. Glücklicherweise war solch eine Arbeit, in Entwicklung. Die daraus entstandene technische Lösung konnte genutzt werden, um die Datenstrukturen, welche im Zuge dieser Arbeit entwickelt wurden, darzustellen. Dies bedurfte jedoch einiger Absprachen und Planung, um die Kommunikation zwischen Datenbank und Webfrontend zu ermöglichen. Dabei lag der Fokus auf der Umsetzung von Kommunikation der Systeme über das Internet und die Definition einer zu verwendeten Datenstruktur. Damit es während der parallelen Entwicklung beider Systeme nicht zu Unstimmigkeiten kam wurde, sich auf eine separate Datenbank mit dem gleichen Aufbau wie die zuvor beschriebene OLTP Struktur geeinigt. Damit stehen stets alle Informationen bereit, welche auf der Webseite dargestellt werden sollen. Nicht nur die Datenstruktur, sondern auch die Kommunikation der Systeme, welche sich auf unterschiedlichen Servern befinden musste festgelegt werden. Die folgende Abbildung zeigt den geplanten Systemaufbau, welcher definiert wurde.

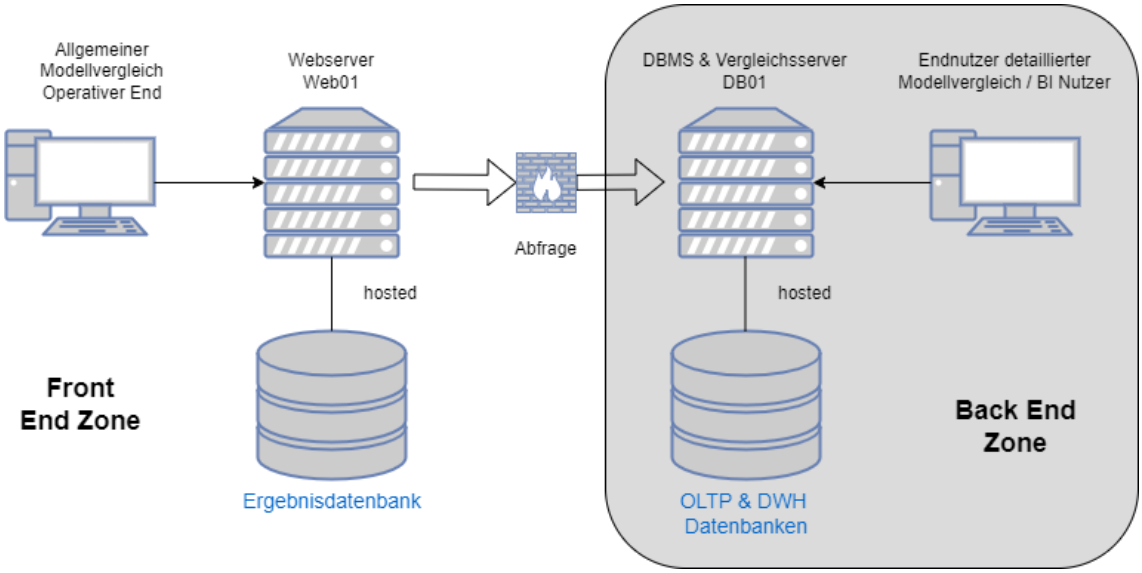

Abbildung 12: Darstellung Systemaufbau

Wie auf Abbildung 12 zu erkennen, stellt der Webserver neben der Website auch eine eigene Datenbank bereit. Diese enthält nur die zuvor erwähnte Kopie der OLTP Datenbank und dient als staging Area, also Bereitstellungsbereich um die abgefragten Daten nicht dauerhaft bereithalten zu müssen. Gleichzeitig wird so auch ermöglicht, dass bei einem Datenbankausfall die Website weiterhin genutzt werden kann, auch wenn die Daten dann nicht mehr aktualisiert werden können. Zudem, da beide Projekte parallel in Entwicklung waren, wird somit eine operative Unabhängigkeit, was Tests und Systemanpassungen betrifft, voneinander sichergestellt. Der Webserver wird mit einem Ubuntu 22.04 LTS betrieben und nutzt einen NGINX Webserver für die Bereitstellung. Als Datenbank-Managementsystem wird auf dem Webserver MySQL eingesetzt. Die Daten werden mittels regelmäßiger Abfrage aus dem Backend abgerufen. Ein direkter Zugriff von der Webanwendung zum Datenbankserver wurde diskutiert und auch erfolgreich implementiert. Welche Variante, ein direkter Datenbank Zugriff oder ein regelmäßiger Abgleich außerhalb der Nutzungszeiten, besser ist, hängt unter anderem von der weiteren Entwicklung des Gesamtprojektes ab. Beide Optionen bieten unterschiedliche Vor- und Nachteile. Der direkte Zugriff bietet stets aktuelle Daten und es kann von der Rechenleistung des Backend Servers für komplexe Abfragen profitiert werden. Bei Einsatz einer eigenen Datenbank ist der Webserver weniger anfällig gegen Netzwerkstörungen und kann eigene Aggregationen sowie Berechnungen vorhalten, welche nicht direkt im Backend gespeichert werden. Nachteilig beim synchronisieren der Daten ist der Zeitverzug und der zusätzliche Aufwand die Ergebnisdatenbank zu pflegen und zu verwalten.

Gegen den direkten Datenbankzugriff spricht die höhere Netzwerkbelastung und das geringere Abstraktionslevel zwischen Front- und Backend. Zum aktuellen Zeitpunkt wird die Variante des direkten Zugriffes auf die Datenbank präferiert. Dank regelmäßigen Absprachen zum Vorgehen und der Umsetzung gab es nur wenige Probleme. Details können der Diplomarbeit von Lukas Möbius entnommen werden [Möb25]. Einige der erzielten Ergebnisse werden im folgenden Abschnitt dargestellt.

4 Entwicklung

4.1 Umsetzung Systemstruktur

Die Umsetzung begann zunächst mit dem Aufbau der grundlegenden Online Transaction Processing (OLTP) Datenbankstruktur, also der Datenbank welche für den aktiven Betrieb vorgesehen ist. Wie zuvor erwähnt ist diese Datenbank zum momentanen Projektstand noch mehr Redundanz als notwendige Systemkomponente, wird aber im weiteren Projektverlauf eine zentrale Rolle einnehmen. Da zum Beispiel eine Implementation in aktive Geschäftsprozesse möglich sein soll, ist es wichtig diese Datenbank als staging area für die Data Warehouse Datenbank zu haben. Die OLTP Datenbank wird im geplanten Betrieb kontinuierlich von den vollständigen Simulationslogs der einzelnen Durchläufe bereinigt, welche in die Datawarehouse Datenbank gespiegelt werden. Für die Umsetzung der OLTP Datenbank wurden die entsprechenden SQL Anweisungen erstellt und auf dem SQL Server ausgeführt. Die verwendeten Befehle können **Anlage 1** entnommen werden. Ebenso wurde die Data Warehouse Datenbank mittels SQL Befehlen angelegt, welche in **Anlage 2** abgebildet sind.
Mit der grundlegenden Datenbankstruktur aufgebaut konnten erste Daten in die einzelnen Tabellen geladen werden. Hierfür wurde zunächst auf die bereits vorhanden Simulationsdaten des *DynPlus* Modells zurückgegriffen, zusammen mit den öffentlich verfügbaren historischen Strompreisdaten. Die historischen Energiebeschaffungskosten konnten durch eine vom Frauenhofer Institute bereitgestellte API bezogen werden [Ene25]. Diese Daten wurden mittels eines Python Skripts abgerufen und anschließend mit einem weiteren Skript in SQL Insert Statements transformiert um sie in die Datenbank laden zu können. Dabei wurden Metadaten und fehlende Werte, wie der gesamte Strompreis und Höhe der Steuern und Abgaben, ergänzt. Wie im Kapitel 2 „Theoretische Vorbereitung “ bereits erwähnt, sind die Abgaben auf dynamische Strompreise je nach Kommune unterschiedlich, nur die Kosten der Energiebeschaffung sind deutschlandweit (bei gleicher Strombörse) gleich. Für den Systemaufbau und Test wurde sich entschieden die Steuern zu approximieren. Ein Netzbetreiber hat natürlich Zugriff auf die genauen örtlichen historischen Steuerdaten, was einen konkreten Vergleich alternativer Preismodelle ermöglichen würde. Für die Approximation der Steuerhöhe wurden zufällige Werte aus einer Normalverteilung der historischen Steuerwerte für den Raum Dresden gezogen. Die folgende Abbildung zeigt die Verteilung:

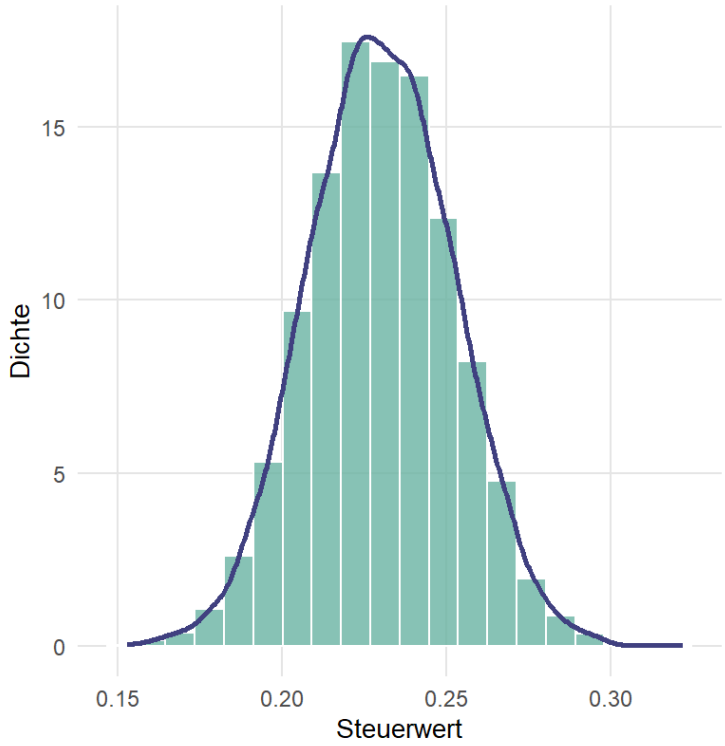

Abbildung 13: Verteilung der generierten Steuerwerte

Es ist wichtig festzuhalten, dass es sich dabei nicht um echte oder besonders realitätsnahe Daten handelt. Die Höhe der Steuer ist abhängig von den Energiebeschaffungskosten, was beim ziehen aus der Normalverteilung nicht berücksichtigt wird. Damit eignet sich die Normalverteilung nur sehr schlecht zum bestimmen einzelner Werte. Der Fehler bei individuellen Werten wird jedoch über eine größere Zeitspanne hinweg ausgeglichen und somit können Testdaten mit realistischen Mustern erzeugt werden. Gleichzeitig kann mittels Anpassung der Standardabweichung Sigma eine Preisanomalie oder Steueranpassung approximiert werden. Für einen echten Entscheidungsvergleich müsste die Normalverteilung durch die Berechnungsvorschrift der Strompreissteuer der jeweiligen Kommune ausgetauscht werden. Da dies im Rahmen der Arbeit zeitlich nicht möglich war, wurde die allgemeine Annäherung mittels der Normalverteilung gewählt. Für das Einfügen in die Datenbank wurden wie zuvor erwähnt kurze Python Skripts entwickelt, welche in **Anlage 3** für das abrufen aus der API [Ene25] und in **Anlage 4** für das Erstellen der *INSERT* Befehle abgebildet sind. Hiermit ist es möglich die historischen Strompreisdaten bereitzustellen. Mit der grundlegenden Datenbankstruktur aufgebaut und den historischen Daten integriert, konnten die SIM_LOGS eingespielt werden.

Dieser Prozess wird sich je nach Modell und Implementation unterscheiden, da die Struktur der Ergebnisdaten variieren kann. Für die vorhandene Implementation des Modells *DynPlus* werden die Ergebnisse in Form einer CSV Datei ausgegeben, diese konnte ebenfalls mittels eines Python Skripts in eine Liste entsprechender *INSERT*-Befehle umgewandelt werden. Das entsprechende Skript kann **Anlage 5** entnommen werden. Bei den simulierten Werten handelte es sich um die stündlichen Preise vom 01.01.2023 0 Uhr bis zum 01.06.2025 12 Uhr. Eine Automatisierung dieses Vorgangs wurde in diesem Fall nicht vorgenommen, da für Testzwecke kein kontinuierlicher Datenfluss bereitstehen musste. Dies ließ sich jedoch mit beispielsweise einem Windows Server Job oder einer programmierten Erweiterung umsetzten. Ebenfalls noch un-automatisiert ist der Extract-Transform-Load-Prozess, da zum Entwicklungszeitpunkt, aufgrund des niedrigen unregelmäßigen Datenaufkommens kein Mehrnutzen darin bestand. Der Microsoft SQL Server bietet hierfür jedoch gute Werkzeuge mit Jobs, welche nach bestimmten Kriterien regelmäßig ausgeführt werden.
Mit den beiden Datenbankstrukturen aufgebaut und mit entsprechenden historischen Strompreisen gefüllt, mussten noch die *SIM-LOGS* der einzelnen Implementationen eingefügt werden. Hierfür standen zwei Implementationen des *Dyn-Plus*-Modells zur Verfügung: Die Originale, vorgestellt in [Wie24], durch den Autor Professor Wiedemann bereitgestellt und eine eigene Nachentwicklung, welche absichtlich vereinzelte Fehler im Basisdatenset beinhaltet. Die Fehler wurden eingebaut um bei der Implementation der Validierung bessere Tests durchführen zu können, sowie anschauliche Beispiele zu produzieren.

4.2 Implementation Validierung

Mit der Vergleichsstruktur geplant und implementiert war der nächste Schritt die Umsetzung der Validierung. Das Ziel hierbei ist es die Unterschiede in verschiedenen Implementationen desselben dynamischen Strompreismodells, sollten solche existieren, sichtbar zu machen. Unterschiede können dabei durch Fehler in der Programmierung, abweichende interne Logik, unterschiedliche Daten für die Stromerzeugungskosten oder Differenzen im Umgang mit Rundung bei verschiedenen Programmiersprachen entstehen. Eine einfache Validierung kann nicht den Ursprung dieser Fehler ermitteln, gibt aber oft ein gutes Indiz auf die Herkunft des Problems. Für eine solche Validierung ist es zunächst notwendig eine der Preismodellimplementationen als Basis zu verwenden, also seine Korrektheit anzunehmen, gegen welches die anderen verglichen werden können. Diese Information, zusammen mit einer zu vergleichenden Zeitspanne, sind die Eingabeparameter für die Validierungsprozedur. Diese wurde auf dem SQL Server, welcher auf dem Datenbank-Server bereitgestellt wird, als Stored Procedure implementiert. Der hierfür notwendige SQL-Code befindet sich in **Anlage 6**. Die Prozedur trägt den Namen *usp_CompareSimPrices* und wird, wie bereits erwähnt, mit den Parametern für den Basisalgorithmus und den zu vergleichenden Algorithmus, jeweils in Form der entsprechenden AlgorithmusID aufgerufen. Zusätzlich wird die zu vergleichende Zeitspanne in Form von Start und Enddatum, beide im DATETIME Format, übergeben. Ein beispielhafter Aufruf der Prozedur findet sich in **Anlage 7**. Für das Speichern der Validierungsergebnisse wird eine Hilfstabelle mit der Bezeichnung SIM_COMPARISON_RESULTS verwendet. Nach einer initialen Plausibilitätsprüfung der Zeitparameter (Startdatum $\leq$ Enddatum) werden alle bestehenden Vergleichseinträge für das angegebene Algorithmuspaar, im gewählten Zeitraum, aus der Tabelle gelöscht. Dies verhindert doppelte Einträge und stellt sicher, dass nur aktuelle Vergleichsdaten vorliegen. Im Anschluss erfolgt der eigentliche Vergleich: Die Tabelle SIMULATION_RESULTS_TEMP, welche die temporären Simulationsergebnisse enthält, wird per Self-Join auf Basis des Zeitstempels Sim_Date verbunden. Dabei werden nur Datensätze berücksichtigt, bei denen beide Simulationen einen gültigen Wert enthalten. Für jedes gemeinsame Zeitintervall wird anschließend die prozentuale Abweichung der simulierten Strompreise berechnet:

$$PercentDifference = (\frac{CompareSimPrice - BaseSimPrice}{BaseSimPrice}) \times 100$$

Sollte der Basispreis Base_Sim_Price den Wert 0 aufweisen, wird die Differenz als NULL gespeichert, um eine Division durch null zu vermeiden. Durch die Verwendung der AlgorithmusID als Fremdschlüssel ist immer eine eindeutige Zuordnung der Vergleichsergebnisse gegeben. Nach erfolgreicher Durchführung wird noch die Anzahl der erzeugten Vergleichsdatensätze ausgegeben. Mit der Validierungsprozedur lässt sich nun die Integrität der Modellimplementationen feststellen und vergleichen. Diese Ergebnisse lassen sich auch gut visualisieren, beispielsweise mittels der dynamischen Darstellung von [Möb25].

Abbildung 14: Validierung Ergebnisbeispiel (angepasste Darstellung)

Die Grafik zeigt den Vergleich zweier *Dyn-Plus* Implementierungen, welche primär gleich zu arbeiten scheinen, jedoch für zwei Datensätze einen sehr starken und einen eher geringeren Unterschied aufweisen. Dies lässt auf einen Fehler in der Datenbasis, also den Stromerzeugungskosten schließen, oder aber einen Rundungsfehler aufgrund unterschiedlicher Behandlung in der Gleitkommaarithmetik. Der starke Unterschied spricht aber eher dagegen und lässt den Datenfehler wahrscheinlicher wirken.

4.3 Aufbau Data Warehouse

Mit den Datenbankgrundlagen aufgebaut und der Validierungsprozedur implementiert war der nächste Schritt der Aufbau des Data Warehouses Cubes. Hierfür wurde, wie zuvor erwähnt Visual Studio 2022 verwendet. Im ersten Schritt wurde ein neues Projekt des Typs *Analysis Services Multidimensional Projects* angelegt, hierfür wird eine entsprechende Erweiterung benötigt [Cor22b]. Mit dem Projekt angelegt ist nächste Schritt den SQL Server als Datenquelle zu verbinden. Dazu muss eine *Data Soruce* (dt. Datenquelle) angelegt werden, welche eine Datenbankschnittstelle verwendet, um mittels Nutzername und Passwort auf den Zielserver zuzugreifen. Der verwendete Nutzer muss Lese- und Schreibrechte auf die Zieldatenbank besitzen. Als Schnittstellentyp wird *Microsoft OLE DB Driver for SQL Server* verwendet. Zudem muss noch der initale Katalog festgelegt werden, dabei handelt es sich um die Datenbank, welche die Tabellen für das Data Warehouse enthält. Mit diesem Objekt lässt sich eine Verbindung zum Datenbankserver herstellen. Der nächste Schritt, das Auswählen der notwendigen Daten, erfolgt mittels eines *Data Source Views* (dt. Datenquellen Sicht). Dieser bestimmt, welche Tabellen als Dimensionen in den Data Warehouse Cube aufgenommen werden. Es wurden die Fakten- sowie Dimensionstabellen, welche in Abbildung 11 dargestellt sind, ausgewählt. Mit der Datenbanksicht angelegt, konnte als letzter Schritt der eigentliche Cube (auch OLAP-Würfel) angelegt werden. Bei diesem handelt es sich um eine multidimensionale Anordung der Dimensionen, welche im relationalen Datenmodell als Tabellen angelegt wurden. Entsprechend wurden erneut die gleichen Tabellen wie in Abbildung 11 dargestellt, ausgewählt. Mit diesen Dimensionen ist eine granulare Analyse der Simulationsergebnisse entlang verschiedener Achsen möglich. Anschließend mit den Vorbereitungen abgeschlossen, konnte das Projekt kompiliert werden. Hierbei werden auch die notwendigen Strukturen auf dem Datenbankserver angelegt. Der entstandene Data Warehouse Cube lässt sich wie folgt darstellen:

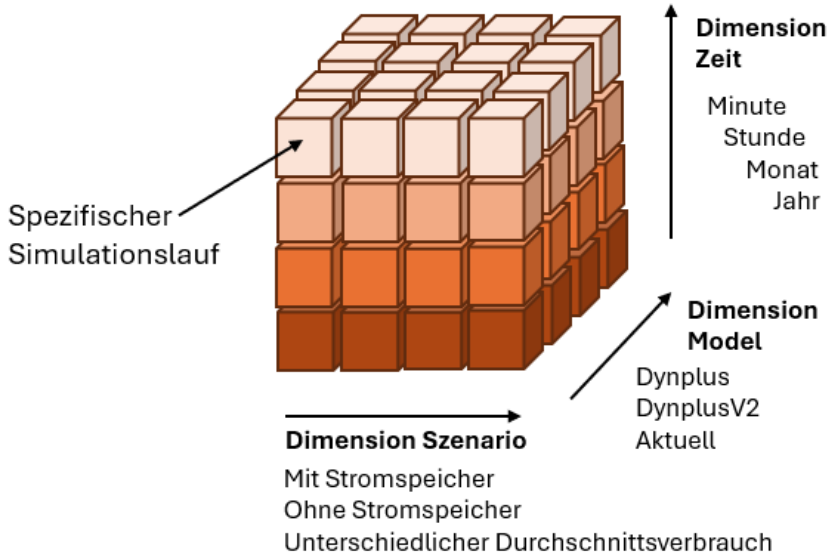

Abbildung 15: Data Warehouse Cube Darstellung

Da sich die sechs Dimensionen des Cubes nicht sinnvoll visualisieren lassen, sind in der Abbildung nur drei dargestellt. Mit dieser Struktur lassen sich verschiedene Operationen durchführen, um Informationen darzustellen. Beispiele hierfür sind *slicing, dicing, pivoting, drill-down oder roll-up.*

- *slicing:* Auswahl einer bestimmten Dimensionsebene.
- *dicing:* Betrachtung eines Teilwürfels durch Einschränkung mehrerer Dimensionen.
- *pivoting:* Verändern der Sicht auf eine andere Dimension.
- *drill-down:* Navigation in detailliertere Datenebenen.
- *roll-up:* Aggregation auf höhere Ebenen.

Alle diese Operationen haben gemeinsam, dass sie die Sicht auf den Würfel verändern, sodass mehr oder weniger Informationen dargestellt werden. Auch lassen sich Aggregation in detaillierte Werte herunterbrechen und wieder verdichten. Die Dimensionen wiederum sind der Kontext sogenannter *Measures*, dies sind meist numerische Daten, welche im Data-Warehouse Cube hinterlegt ist. Ein *Measure* ist beispielsweise ein simulierter Strompreis, welcher erst im Kontext mit der Dimension Zeit (TIME) und der Dimension Simulationsmodel (SIM_MODEL) sinnvoll ausgewertet werden kann. *Measures* sind oft das Ziel von Analysen und Auswertungen und können mittels Multidimensional Expressions (im folgenden MDX) abgefragt werden [Cor22c]. Sie müssen neben den Dimensionen ebenfalls im Data Warehouse Cube definiert werden.

4.4 Einrichten der BI Funktionen

Mit der erfolgreichen Umsetzung des Data-Warehouse-Cube (vgl. Kapitel 4.3) wurde die Grundlage geschaffen, um verschiedene Business Intelligence-Funktionen auf die simulierten sowie historischen Strompreisdaten anzuwenden. Ziel ist es, Entscheidungsträgern eine explorative und analytische Auswertung der Daten zu ermöglichen, ohne, dass dabei tiefgreifende technische Kenntnisse vorausgesetzt werden. Die Analyse erfolgt entlang der im Cube definierten Dimensionen (vgl. Abbildung 15). Im Folgenden werden vier exemplarische BI-Analysen vorgestellt, die auf dem entwickelten System theoretisch umsetzbar sind. Die Implementation erfolgt mittels MDX, der Abfragesprache für OLAP-Cubes im Microsoft SQL Server Analysis Services (SSAS) [Cor22c] [GC06].

1. Durchschnittlicher Strompreis pro Modell und Monat Diese Analyse erlaubt es, die durchschnittlichen Strompreise je Preismodell über einen Monat hinweg zu vergleichen. Dadurch lassen sich saisonale Schwankungen und Modellunterschiede identifizieren.

```
--Average powerprice per modell and month
SELECT
{[Measures].[AveragePrice]} ON COLUMNS,
NON EMPTY
([TIME].[Month].Members * [SIM_MODEL].[algorithm].Members) ON ROWS
FROM [SimCube]
WHERE ([SCENARIO].MEMBERS)
```

2. Ersparnis gegenüber Fixpreis-Tarif Zur Bewertung der Wirtschaftlichkeit eines dynamischen Strompreismodells kann es sinnvoll sein, die Differenz zum durchschnittlichen festen Strompreis zu berechnen.

```
--Savings dynamic price
WITH MEMBER [Measures].[SavingsVsFixed] AS
([Measures].[FixedPrice]) - ([Measures].[Dyn_Price])
SELECT
{[Measures].[SavingsVsFixed]} ON COLUMNS,
[SIM_MODEL].[algorithm].Members ON ROWS
FROM [SimCube]
```

3. Finden von Hochpreisintervallen Die Erkennung von Zeiträumen mit besonders hohen Strompreisen kann sinnvoll sein, z.B. zur Bewertung der Resilienz eines Modells gegenüber Dunkelflauten. Sie kann auch zur Optimierung von Speicherstrategien genutzt werden.

```
-- Highpricezone DynPlus
SELECT
{[Measures].[AveragePrice]} ON COLUMNS,
FILTER(
[Time].[Hour].Members,
[Measures].[AveragePrice] > 0.30
) ON ROWS
FROM [SimCube]
WHERE ([SIM_MODEL].[algorithm].[DynPlus])
```

4. Modellvergleich nach Abgabenlast Ein zentrale Komponente bei einem alternativen Strompreismodell ist die Verteilung der Abgabelast. Es sollte nach Möglichkeit kein Verlust für stattliche Akteure entstehen.

```
-- Compare taxes all models 2025
SELECT
{[Measures].[TotalTaxes]} ON COLUMNS,
[SIM_MODEL].[algorithm].Members ON ROWS
FROM [SimCube]
WHERE ([Time].[Year].[2025])
```

Diese vier Beispiele zeigen, wie das entwickelte System, verwendet werden kann um die Entscheidungsfindung zu unterstützen. Es sind jedoch eine Vielzahl weiterer Analysen denkbar, wie etwa zur Optimierung Stromspeicherausnutzung oder einer Standortbewertung abhängig von der Kommune.
Zur Auswertung der im Data-Warehouse-Cube gespeicherten Daten können verschiedene Werkzeuge verwendet werden. Zwei davon eignen sich besonders für den in der Arbeit gewählten Aufbau mittels Microsoft SQL-Server: Microsofts Excel und Microsofts Power BI. Beide Anwendungen bieten eine Integration mit dem SQL Server Analysis Services (SSAS) und ermöglichen eine direkte Verbindung zum OLAP-Cube.

Excel ermöglicht dies über die Funktion *Daten abrufen* → *Aus Datenbank* → *Von Analysis Services*. Hiermit kann eine Verbindung zum Cube hergestellt werden. Nach Eingabe der Serveradresse und Auswahl der Datenbank kann der Data-Warehouse-Cube als PivotTable eingebunden werden. Die Dimensionen und Measures des Cubes stehen anschließend in der Feldliste zur Verfügung und können in Zeilen, Spalten und Wertebereiche angeordnet werden. Dadurch lassen sich interaktive Pivot-Analysen erstellen, um beispielsweise die Darstellung von Strompreisentwicklungen über die Zeit oder den Vergleich von Modellen hinsichtlich ihrer Abgabenlast zu ermöglichen. Microsoft Power BI wiederum bietet eine noch leistungsfähigere Umgebung für die visuelle Analyse. Über die Datenquelle *SQL Server Analysis Services* kann der Cube im *Live-Verbindungsmodus* eingebunden werden. Dies ermöglicht eine direkte Abfrage des Cubes ohne Datenreplikation. Die Measures und Dimensionen werden automatisch erkannt und stehen für die Erstellung vom Dashboards zur Verfügung. Power BI erlaubt darüber hinaus die Kombination mit weiteren Datenquellen, zum Beispiel Wetterdaten oder Verbrauchsstatistiken, um komplexe Zusammenhänge visuell aufzubereiten.
Beide Werkzeuge unterstützen die zuvor beschriebenen MDX-basierten Analysen. In Power BI können diese über benutzerdefinierte Measures im DAX-Format nachgebildet werden, während Excel direkt MDX-Abfragen über den PivotTable-Editor oder das PowerPivot-Modul unterstützt. Somit ist eine explorative Analyse der verschiedenen Simulationsergebnisse durch technische und nicht-technische Stakeholder möglich. Besonders hervorzuheben ist die Möglichkeit, durch *Drill-Down* und *Slicing* gezielt in bestimmte Zeiträume, Modelle oder Szenarien zu navigieren und so fundierte Entscheidungen auf Basis der simulierten Daten zu treffen.

4.5 Datenauswertung

Für die Auswertung der Daten, können neben den zuvor erwähnten Datenbank- und Validierungsfunktionen auch verschiedene BI Werkzeuge verwendet werden. Mit diesen ist eine visuelle Auswertung der Simulationsergebnisse für verschiedene Zielgruppen möglich. Aber auch die in Abschnitt 3.4 vorgestellte dynamische Darstellung mittels der Arbeit von [Möb25], kann verwendet werden. Diese Anwendung ist besonders gut geeignet um Interessierten die Zusammenhänge dynamischer Strompreise darzustellen, ohne dafür ein BI-Werkzeug zu verwenden. Hierfür greift die Webanwendung direkt auf die Datenbank zu, um verschiedene Informationen abzufragen und zu aggregieren. Historische, sowie simulierte Daten lassen sich als Zeitreihen oder Balkendiagramm darstellen und visuell vergleichen. Dabei lassen sich auch verschiedene Werte filtern um die Ansicht zu verfeinern.

Die folgende Grafik zeigt eine Darstellung der Stromerzeugungskosten, die kumulativen Abgaben und Steuern und den gesamten Strompreis für einen gewissen Zeitraum, zusammen mit dem Strompreis nach *Dyn-Plus* Preismodell für den selben Zeitraum.

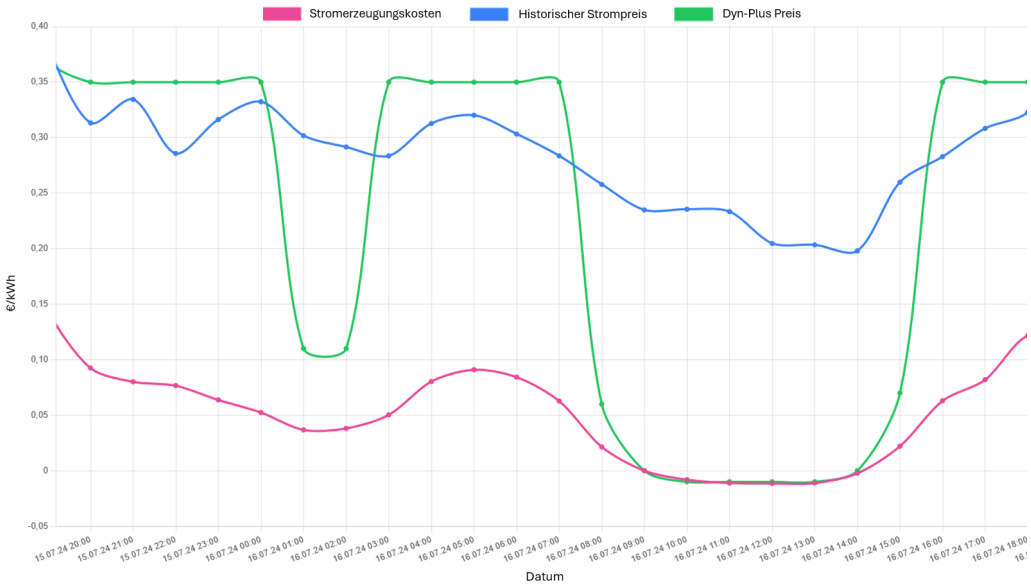

Abbildung 16: Vergleich Historische Strompreise mit *Dyn-Plus* Simulationsergebnissen (angepasste Darstellung)

In diesem Beispiel lässt sich sehr gut die Ausgleichsoperation von *Dyn-Plus* erkennen, wobei zuvor nicht erhobene Abgaben auf Mittelpreise umgelegt werden. Die Website bietet dabei die Möglichkeit verschiedene Modelle in einem frei wählbaren Zeitraum miteinander zu vergleichen. Auch können die Ergebnisse der Validierungsfunktion (vgl. Kapitel 4.2) in Form von zwei verschiedenen Ansichten dargestellt werden.

4.6 Systemtest

Abschließend sollten die entwickelten Systeme noch einem Funktionstest unterzogen werden. Das Ziel dabei war es, die grundlegende Funktionsfähigkeit der Datenaufnahme, Verarbeitung, Validierung und Auswertung sicherzustellen. Da viele Komponenten im Verlauf der Arbeit iterativ entwickelt und direkt zur Umsetzung der nächsten Projektphase verwendet wurden, erfolgte der Systemtest nicht als eigne Phase, sondern fand begleitend zur Entwicklung statt. Im Fokus stand zunächst die Überprüfung der Datenflüsse zwischen den Simulationsmodellen, der OLTP-Datenbank und dem Data Warehouse.

Dabei konnten erfolgreich sowohl historische Strompreisdaten als auch simulierte Werte aus verschiedenen Modellen aufgenommen, verarbeitet und in strukturierter Form gespeichert werden.
Die Validierungsprozedur (vgl. Abschnitt 4.2) wurde mit zwei Implementationen des DynPlus-Modells getestet, darunter einer absichtlich fehlerhafte Variante. Die Ergebnisse zeigten, dass Abweichungen in den Ergebnissen zuverlässig erkannt und quantifiziert werden können. Ein weiterer Schwerpunkt lag auf der Nutzung von verschiedenen BI-Werkzeuge mit dem entwickelten Data-Warehouse-Cube. Die Anbindung an Microsoft Excel und Power BI (vgl. Abschnitt 4.5) wurde beschrieben und konnte für beide auch erfolgreich getestet werden. In beiden Anwendungen standen die definierten Dimensionen und Kennzahlen zur Verfügung und konnten für Analysen genutzt werden. Auch die dynamische Webanwendung, welche in im Zusammenhang mit der Diplomarbeit von Herrn Möbius [Möb25] entstand, wurde erfolgreich mit der OLTP-Datenbank verbunden. Die Kommunikation zwischen Backend und Frontend wurde dabei umgesetzt, wobei sowohl ein direkter Datenbankzugriff als auch ein synchronisierter Datenabgleich implementiert und getestet wurden. Die Darstellung der Simulationsergebnisse auf der Website ermöglichte eine explorative Analyse und erlaubt eine Darstellung der Ergebnisse dieser Arbeit außerhalb von Fachanwendungen. Trotz der erfolgreichen Tests ist anzumerken, dass das System zum aktuellen Zeitpunkt noch nicht für einen produktiven Einsatz geeignet ist. Beispielsweise fehlen noch Prüfmechanismen für Eingabedaten, eine automatisierte Fehlerbehandlung sowie ein durchgängiges Monitoring der Datenqualität. Auch der ETL-Prozess zur Überführung der Daten von der OLTP-Datenbank in das Data Warehouse wurde bislang nur manuell durchgeführt. Für einen operativen Einsatz wären hier automatisierte Jobs und Validierungsroutinen erforderlich. Nichtsdestotrotz konnte im Rahmen dieser Arbeit gezeigt werden, dass alle für die Beantwortung der Forschungsfrage relevanten Funktionen erfolgreich umgesetzt wurden. Die entwickelte Systemarchitektur ist modular aufgebaut und erlaubt eine schrittweise Erweiterung, um beispielsweise neue Preismodelle, zusätzliche Dimensionen (wie Geografie oder Verbrauchsverhalten) oder externe Datenquellen. Ein intensiver End-to-End-Test unter realen Bedingungen wäre vor einer produktiven Nutzung in einem wirtschaftlichen Kontext empfehlenswert.

5 Diskussion

5.1 Bewertung des Modells als Entscheidungsgrundlage

Das entwickelte System bietet eine fundierte Entscheidungsgrundlage für die Bewertung alternativer Strompreismodelle und deren Implementationen. Durch die Möglichkeit, verschiedene Modelle unter identischen Rahmenbedingungen auszuwerten, können Entscheidungsträger fundierte Aussagen über deren wirtschaftliche Auswirkungen treffen. Dennoch zeigen sich sowohl Stärken als auch Schwächen, die im Folgenden gegenübergestellt werden:

Vorteile

Vergleichbarkeit der Modelle: Durch den gewählten Aufbau mit getrennter OLTP- und DWH-Struktur können Ergebnisdaten harmonisiert und unterschiedlich aufgebaute Simulationsmodelle vergleichbar gemacht werden.

Validierungsmechanismus: Es besteht die Möglichkeit, verschiedene Implementationen desgleichen Modells gegeneinander zu vergleichen. Somit lassen sich Rechen-, Daten oder Rundungsfehler schnell und effektiv finden.

Integration BI-Werkzeuge: Mit der Möglichkeit Programme wie Microsofts Excel oder Power BI anzubinden, können eine Vielzahl von Analysen und Auswertungen auf den Daten durchgeführt werden.

Modularität und Anpassbarkeit: Die Systemarchitektur ist so gestaltet, dass Erweiterungen mit neuen Modellen, Szenarien oder Dimensionen sich relativ einfach umsetzten lassen.

Nachteile

Wenige reale Daten: Da alternative Strompreismodelle wie *Dyn-Plus* noch nicht in der Praxis umgesetzt sind, fehlen empirische Daten um die Simulationsergebnisse mit realen Umständen zu vergleichen. Somit sinkt auch die Sicherheit der durchgeführten Analysen.

Begrenzte Modellanzahl im Test: Die praktische Umsetzung beschränkte sich auf wenige Modelle mit gleicher Zeitsegmentierung. Die theoretische Unterstützung unterschiedlicher Zeitaufteilungen konnte nicht getestet werden.

Abhängigkeit vom Microsoft-Ökosystem: Die Wahl von SQL Server, SSAS und Visual Studio führt zu einer starken Bindung an die Software eines einzelnen Herstellers. Dies kann in produktiven Umgebungen zu hohen Lizenzkosten führen und schränkt die Plattformunabhängigkeit ein.

Prüfmechanismen für Eingabedaten: Momentan werden Simulationsdaten noch nicht auf Plausibilität geprüft. Mit der aktuellen Ausrichtung als Forschungssystem ist dies jedoch von geringer Auswirkung. Es kann davon ausgegangen werden, dass alle Systemteilnehmer zweckmäßige Daten bereitstellen wollen.

Das System bietet eine solide Grundlage für die Analyse und den Vergleich alternativer dynamischer Strompreismodelle. Es eignet sich besonders für Forschung und Entwicklung von Prototypen. Hervorzuheben ist die Fähigkeit des Systems, Unterschiede in der Steuerverteilung bei einzelnen Modellen und deren Auswirkungen auf Endverbraucher transparent darzustellen. Für den operativen Einsatz in der Energiewirtschaft sind jedoch noch technische Erweiterungen empfehlenswert.

5.2 Technische Alternativen

Im Zuge der Arbeit wurden auch technische Alternativen betrachtet, welche aus verschieden Gründen jedoch nicht eingesetzt wurden oder konnten. Aufgrund von zeitlichen Beschränkungen war es leider nicht möglich, die verschiedenen Lösungen detailliert zu vergleichen, weswegen nur ein oberflächlicher Vergleich aufgrund von Datenblättern und Dokumentationen durchgeführt wurde. Es besteht die Möglichkeit, dass die in dieser Arbeit beschrieben Modelle sich in einer anderen technischen Lösung besser abbilden ließen. Allen voran muss der Fokus auf Produkte des Herstellers Microsoft angemerkt werden und die daraus resultierende Abhängigkeit vom Windows-Ökosystem. Sollte die in dieser Arbeit entwickelte Vergleichs- und Validierungsplattform in einem wirtschaftlichen Kontext genutzt werden, entstehen nicht zu vernachlässigende Lizenzkosten. Eine alternative Umsetzung mittels OpenSource Lösungen unter Linux wurde kurzzeitig in Erwägung gezogen, jedoch Zeitnahe verworfen. Zwar gibt es verschiedene DBMS unter Linux, welche für diese Arbeit hätten verwendet werden können, aber das Fehlen eines Äquivalents zum Analysis Service des SQL Servers war ausschlaggebend, sich gegen diese zu entscheiden. Da der Analysis Service die technische Basis zum Erstellen und Nutzen des Data Warehouse Cubes darstellt, war er ein zwingender Bestandteil der Arbeit. Es existieren Softwareprodukte, um einen Cube unter Linux zu erstellen, aber da keinerlei Erfahrungen mit diesen Werkzeugen bestand, wurde sich für das Microsoft Produkt entschieden, da hier bereits technische Vorerfahrung bestand. Eine Folgearbeit könnte die Praktikabilität von OpenSource Produkten zur Umsetzung der hier dargestellten Funktionen evaluieren. Aufgrund der Abhängigkeit vom Analysis Service war auch der Einsatz von Visual Studio notwendig, da es sich dabei um die vom Hersteller vorgesehene Entwicklungsumgebung handelt.

Es wurden keine technischen Alternativen für die Entwicklungsumgebung oder die verwendeten Programmiersprachen betrachtet, aufgrund der starken technischen Abhängigkeit der einzelnen Produkte.
Der in Kapitel 2.4 erwähnte Ansatz, die Lösung in der Azure Cloud umzusetzen, hätte zusätzliche technische Möglichkeiten geboten. Beispielweise die direkte Integration in eine PowerBI Umgebung bei entsprechender Lizenzierung des Tenants oder auch Werkzeuge wie PowerAutomate um Berichterstellung und Reporting zu automatisieren. Zugleich bietet ein Azure Tenant die Möglichkeit als Schnittstelle zwischen vielen verschieden Geschäftsanwendungen zu dienen. Die Planung hierfür war schon soweit fortgeschritten, dass bereits eine Auflistung der notwendigen Azure Komponenten zur Umsetzung der Systemstruktur erstellt wurde. Jedoch waren die damit verbundenen monatlichen Azure-Kosten zu hoch für den Projektrahmen und daher wurde diese Lösung verworfen.

Datenbankserver Umgesetzt mit einer D4v5 VM (2 vCPUs 8 GB RAM), mit 512 GiB SSD Standard Tier als Managed Disks. Dafür entstehen monatliche Kosten von 143,41€ ohne Lizenzgebühren oder 1.193,39 € mit Lizenzen für das Betriebssystem und den SQL Server.

Webserver Umgesetzt mit einer B2ms (2 vCPUs 8 GB RAM), mit 64 GiB SSD Standard Tier als Managed Disks. Dafür entstehen monatliche Kosten von 63,96 €, da es sich um ein Linux System handelt werden keine Lizenzgebühren erhoben.

Firewall Umgesetzt mit einer Azure Firewall im Basic Tarif, hierdurch würden monatliche Kosten von 249,06 € entstehen.

Standort Aus Gründen des Datenschutzes wurden alle Komponenten am Standort Germany West Central geplant und dies mit einer durchgehenden Betriebszeit von 730 Stunden.

Diese Lösung hätte somit monatliche Kosten von mindestens 489,22€ undbis zu 1.539,20€, je nachdem, ob Lizenzen für den Windows- und SQL-Server gebucht werden müssen oder nicht. Cloud-übliche Preisfluktuationen von bis zu 10% wären zu erwarten. Die Kosten wurden ermittelt und können mit dem Azure Preisrechner nachvollzogen werden [Azu25], wobei die Preise zum Abgabedatum der Arbeit zuletzt überprüft wurden und möglichen Änderungen unterliegen.

Anmerkung des Autors: "Die hier erwähnten Azure Komponenten entsprechen den gängigen Empfehlungen seitens Microsoft für die vorgesehen Anwendungsfälle. Es ist die Meinung des Autors basierend auf Praxiserfahrung, dass auch deutlich kleiner skalierte Maschinen hierfür eingesetzt werden könnten. Auch ist der Standort Germany West Central aus Kostensicht nicht ideal. Solange ein Land in der Europäischen Union als Standortanforderung ausreichend ist, wären die Rechenzentren *North Europe* oder *West Europe* zu bevorzugen."

5.3 Robustheit und Anpassbarkeit

Im Kontext dieser Arbeit bedeutet Robustheit, wie schwerwiegend sich Änderungen auf das Gesamtsystem auswirken und wie hoch der Aufwand zur Umsetzung von Anpassungen relativ zu deren Umfang ist. Ein Beispiel für die Robustheit ist die Validierungslogik, die in Kapitel 4.2 beschrieben wurde. Durch den Vergleich mehrerer Implementierungen desselben Modells konnte sichergestellt werden, dass Rechenfehler oder Rundungsdifferenzen frühzeitig erkannt und behoben werden. Die Möglichkeit, verschiedene Implementierungen gegeneinander zu testen, erhöht nicht nur die Verlässlichkeit der Ergebnisse, sondern erlaubt auch eine kontinuierliche Qualitätskontrolle bei der Integration neuer Modelle. Zwar wurde das System mit Anpassbarkeit als eine der Grundanforderungen entworfen, jedoch gilt dies nicht für alle Komponenten. Die Datenbanken lassen sich leicht um zusätzliche Tabellen erweitern, jedoch kann eine häufige Anpassung des Data Warehouse Modells zu hohen Aufwänden führen. Sollte dies in Zukunft häufiger im Zusammenhang mit einem hohen Neudaten aufkommen auftreten, wäre eine Neumodellierung sinnvoll.
Durch den Aufbau des Data Warehouses als Data Vault ließen sich diese Anforderungen besser abdecken, da diese eine hohe Flexibilität bietet. Der Data Vault Ansatz sieht vor, Informationen in drei verschiedenen Kategorien von Tabellen zu speichern und bei Änderungen nicht bestehende Tabellen anzupassen, sondern neue hinzuzufügen. Diese Kategorien sind Hubs, Links und Satelliten. Dadurch kann besser auf sich häufig ändernde Anforderungen reagiert werden.

Hubs speichern die eindeutige Stammdaten (z.B. Kundennummern) und dienen als Sammelpunkt für Daten aus verschiedenen Quellen

Links modellieren die Beziehungen zwischen den Hubs

Satelliten enthalten die beschreibenden Attribute und deren Historie

Mit dieser Struktur ist eine vollständige Historisierung der Daten möglich und zusätzlich lassen sich Ladeprozesse parallelisieren [LO15]. Im Kontext dieser Arbeit könnte der Einsatz eines Data Vault-Modells insbesondere dann sinnvoll sein, wenn eine langfristige Erweiterung oder Umstrukturierung des entwickelten Systems geplant ist.
Die Anpassbarkeit zeigt sich insbesondere in der Fähigkeit des Systems, neue Simulationsmodelle mit abweichender Struktur aufzunehmen. Wie in Kapitel 2.5 erläutert, unterscheiden sich die Modelle nicht nur in ihrer internen Logik, sondern auch in der Struktur ihrer Ergebnisdaten. Durch die flexible Gestaltung der Datenbanktabellen können neue Modelle ohne tiefgreifende Änderungen am Gesamtsystem integriert werden.

5.4 Einschränkungen (Limitations)

Wie jede wissenschaftliche Arbeit unterlag auch diese externen Einschränkungen. Allen voran einer zeitlichen Begrenzung, weswegen nicht alle angesprochen Aspekte vollständig beleuchtet werden konnten. Als disziplinübergreifendes Projekt zeigten sich während der Bearbeitung auch Punkte, zu deren Lösung Fachwissen initial fehlte und aufgebaut werden musste. Was aber nicht für alle Themen, im gegeben Zeitnahmen, vollumfänglich möglich war. Besonders sind die Themenkomplexen des Steuerrechts, sowie den Aufbau des Stromnetzes aus regulatorischer und operativer Sicht zu nennen. Außerdem fehlt die Betrachtung der Rechtslage aus europäischer Sicht, was Auswirkungen auf die tatsächliche Umsetzbarkeit alternativer Strompreismodelle haben könnte. Diese kann nicht abgeschätzt werden. Eine zusätzliche Betrachtung durch Sachverständige dieser Bereiche wäre sinnvoll gewesen. Es existieren auch persönliche Einschränkungen, allem voran ist der Autor überzeugt von der Sinnhaftigkeit und den wirtschaftlichen Vorteilen des *DynPlus* Strompreismodells, es wurde aber nach eine neutrale Darstellung bestem Wissen und Gewissen geachtet. Das aufgebaute System soll eine offene Plattform darstellen, welche Simulationsergebnisse unabhängig von der Struktur des Simulationsmodells aufnehmen, verarbeiten und aggregieren kann. Zum aktuellen Zeitpunkt existieren noch nicht genügend verschiedene Modelle um diesen Aspekt umfangreich testen zu können. Besonders der Punkt über Modelle mit anderen Zeitsegmenten konnte nur theoretisch, aber nicht praktisch evaluiert werden, da alle verwendeten Modelle auf der selben Zeitsegmentierung beruhen. Auch der aktuell viel diskutierte Punkt der generativen KI Modelle muss Erwähnung finden.

In dieser Arbeit wurde kein inhaltlicher Text, welcher einem KI Modell entstammt, verwendet. Ein generatives Modell, Chat GPT-4o von OpenAI wurde lediglich für die Erstellung und Fehlersuche der verschiedenen Skripte, welche für diese Arbeit entwickelt wurden eingesetzt. Das generieren der SQL-Befehle, zum Umsetzten der Datenbankmodelle, war dabei besonders zeitsparend. Die selbständig erstellten ERM-Modelle konnten dem KI-Modell übergebenen werden, welches daraus entsprechende CREATE-Befehle erstellen konnte. Dadurch wurden Tippfehler vorgebeugt und Zeit bei der wiederholten Anpassung der Tabellenstruktur gespart. Sämtlicher generierter Code wurde vor der Ausführung auf Korrektheit überprüft und gegeben falls angepasst. Es wurde auch versucht das KI-Modell als Redakteur, zu verwenden um Fehler bei der Rechtschreibung und Grammatik zu korrigieren. Hierfür wurden zunächst sämtlicher Text in Eigenarbeit erstellt und dann dem KI-Modell stückweise übergeben, unter der Instruktion ausschließlich Fehler zu beheben und keine Inhaltlichen Änderungen vorzunehmen. Hier ließ sich nur ein mäßiger Erfolg verzeichnen, zwar eignet sich die Arbeitsweise eines *Large Language Models* besonders gut für das Korrigieren von langen Texten, aber es ist nur sehr schwer möglich inhaltliche Änderungen seitens des Modells vorzubeugen. Daher mussten alle Texte nochmals überprüft und Satz für Satz verglichen werden. Ein Modell welches nur für diese Aufgabe trainiert ist, wäre in diesem Fall einem allgemeinen Modell wie ChatGPT vorzuziehen gewesen. Auch wenn Quellen sorgfältig ausgewählt wurden und viele der wissenschaftlichen Methodik folgen, kann nicht ausgeschlossen werden, das einzelne Webquellen generative KI zur Inhaltserstellung eingesetzt haben. Diesbezüglich wurde keine Prüfung durchgeführt.
Eine weitere Einschränkung war die Verfügbarkeit von Informationen bezüglich der Berechnung und Historie von Strompreisen. Da, wie zuvor erwähnt, jede Kommune ihre eigenen Berechnungsvorschriften für die Stromsteuer hat, kann der aktuelle Ist-Zustand nur sehr aufwendig abgebildet werden. Zudem bestehen Probleme bei den historischen Strompreisdaten. Zwar sind diese über die API-Schnittstelle des Frauenhofer Institutes verfügbar, aber leider gibt es darin vereinzelte Fehler. Die Schnittstelle gibt Strompreise als Euro je Megawattstunde zurück und rundet dabei auf zwei Nachkommastellen, wobei sehr selten einzelne Stundenwerte deutlich mehr aufweisen. Ein Beispiel dafür wäre der folgende Wert *-0.6000000000000001*. Es handelt sich hierbei wahrscheinlich um einen Fehler in der Gleitkommaberechnung. Dieser tritt in den Daten jedoch sehr selten auf. Eine Überschlagsrechnung ergibt eine Wahrscheinlichkeit von ca. 0,003% das ein Wert in dieser Form auftritt. Die Handhabung dieser Werte in unterschiedlichen Implementationen, könnte zu Folgefehlern in der Simulation führen, welche sich nur schwer nachvollziehen ließen.

6 Abschluss

6.1 Fazit der Arbeit

Die vorliegende Arbeit zeigt, dass ein systematischer Vergleich von alternativen dynamischen Strompreismodellen nicht nur theoretisch, sondern auch praktisch umsetzbar ist. Durch eine fundierte Vorbereitung (vgl. Kapitel 2), einen strukturierten Systementwurf (vgl. Kapitel 3) und einer flexiblen technischen Umsetzung (vgl. Kapitel 4) konnte ein Framework geschaffen werden, welches sich für sowohl wissenschaftliche als auch wirtschaftliche Anwendungen eignet. Die Möglichkeit, verschiedene Implementationen des gleichen Strompreismodells miteinander zu vergleichen, ist eine zentrale Fähigkeit des Systems und die Organisation der Daten in einem Data Warehouse eröffnet die Möglichkeit von Datamining und umfassenden Analysen. Dabei besteht eine konsequente Trennung von operativen und analytischen Daten. Ebenso sind reale Strompreisdaten integriert und können kontinuierlich abgerufen werden. Zusätzlich wurden Schnittstellen für eine dynamische Visualisierung der Daten geschaffen und erfolgreich implementiert, vgl. Kapitel 3.4. Damit leistet die Arbeit einen Beitrag zur Weiterentwicklung dynamischer Preismodelle und bietet eine fundierte Grundlage für weiterführende Forschung und Anwendung. Insgesamt konnte gezeigt werden, dass durch eine sorgfältige Kombination aus Simulation, Datenmodellierung und Business Intelligence ein leistungsfähiges System zur Bewertung dynamischer Strompreise geschaffen werden kann.
Im Zuge der Arbeit eröffneten sich jedoch auch weitere Ziele, welche nicht mehr in diesem Rahmen umgesetzt wurden. So ist die Modellierung des Data Warehouses für den momentanen Anwendungsfall ausgelegt, es hat sich jedoch gezeigt, dass weitere Dimensionen wie beispielsweise für Geografie oder Simulations- Variablen sinnvoll wären. Es wurde sich dagegen entschieden, diese kurzfristig einzubauen, aufgrund von zeitlichen Beschränkungen. Auch konnten BI Funktionalitäten nicht zum gewünschten Maße hin ausgebaut werden, da zum Zeitpunkt der Durchführung nicht genügend Daten zur Verfügung standen.

6.2 Vergleich zur Zielstellung

Abschließend soll die initial gestellte Forschungsfrage erneut betrachtet werden *„Wie lassen sich verschieden strukturierte Simulationsmodelle zielführend und wirtschaftlich effektiv miteinander vergleichen.“* Diese Frage konnte im Rahmen der Arbeit weitgehend beantwortet werden. Dies wurde erreicht durch die Entwicklung und Implementierung eines anpassbaren datenbankgestützten Systems, welches die Ergebnisse von verschiedenen Simulationsmodellen vergleichbar macht und gleichzeitig die Auswertung mittels Business Intelligence Werkzeugen ermöglicht. Die zentrale Herausforderung bestand dabei in der Harmonisierung der Eingangs- und Ergebnisdaten, ohne die Modelle selbst zu verändern. Dies wurde durch die Einführung einer OLTP-Datenbank zur temporären Speicherung und Validierung sowie eines Data-Warehouse-Cubes zur multidimensionalen Analyse gelöst. Auch wurden verschiedene Umsetzungsplattformen betrachtet nach wirtschaftlichen Gesichtspunkten bewertet.

Wie zuvor bereits erwähnt, eröffneten sich noch weitere Ziele zur Erweiterung und Ausbau des Vergleichssystems, welche nur theoretisch beschrieben wurden. Auch gestaltete sich der Projektverlauf nicht wie initial erwartet und es gab weniger Testdaten als erhofft. Dennoch wurde die Systemstruktur so entworfen, dass Modelle mit abweichenden Zeitintervallen unterstützt sind. Ebenso konnte die geplante Erweiterung um die Dimension Geografie und das Einbinden externer Einflussfaktoren nur konzeptionell vorbereitet, aber nicht implementiert werden. Es wurde angenommen, dass weitere Modelle und Implementationen von Projektinteressenten erstellt werden würden. Dies ist zwar der Fall, aber zeitliche Beschränkungen verhinderten die Aufnahme in diese Arbeit. Nichtsdestotrotz wurde eine robuste Struktur entwickelt, welche es erlaubt, dynamische Strompreismodelle mit verschieden Zeitsegmentierungen zu vergleichen und unterschiedliche Implementationen des gleichen Modells gegeneinander zu validieren. Die entwickelte Architektur bietet eine robuste Grundlage für weiterführende Forschung und eine mögliche produktive Anwendung in energiewirtschaftlichen Entscheidungssystemen.

6.3 Ausblick & Folgeentwicklung

Die Ergebnisse dieser Arbeit bieten viele Startpunkte für weitere Forschung und Folgeentwicklungen. Vor allem die Erweiterung des Datenpools um neue alternative Strompreismodelle bietet sich an. Auch ließen sich neue Auswertungs- und Business Intelligence-Funktionen leicht hinzufügen. Eine weitere Möglichkeit, neue Modelle zu erhalten, wäre eine Folgeentwicklung, basierend auf dem *DynPlus* Modell, welche noch mehr Variablen, wie beispielsweise Umwelteinflüsse, saisonale Besonderheiten oder Klimadaten integriert.
Die wohl ausschlaggebendste Erweiterung wäre die Integration einer neuen Dimension für die Geografie beziehungsweise dem Standort. Da die Abgaben sich für bestimmte Energiebeschaffungskosten je nach Kommune in Deutschland bis zu 30% unterscheiden können, wäre dies für eine genaue Standortsimulation unabdinglich. Momentan findet eine deutschlandweite Betrachtung statt, womit die Unterschiede teilweise ausgeglichen werden. Mit einer Erweiterung um den Standort ließe sich detaillierte Szenarien abbilden und auch wenn es kein Bestandteil dieser Arbeit ist, lässt sich annehmen, dass Strompreismodelle außerhalb Deutschlands ebenfalls regionale Unterschiede vorweisen. Zu diesem Zweck müsste die OLTP und DWH - Datenbanken um eine entsprechende Tabelle erweitert werden, um daraus eine neue Dimension zu definieren. Mit diesen Anpassungen ließen sich ebenfalls weitere Filter auf der Webseite für die dynamische Darstellung einfügen. Unabhängig von der Geografie gibt es weitere Faktoren, welche in das Vergleichssystem aufgenommen werden können, vor allem im Zusammenhang mit internationalen Vergleichen.
An dieser Stelle soll auch noch erwähnt werden, dass im parallelen Verlauf zu dieser Arbeit zwei Paper publiziert worden, welche die Entwicklung des *Dyn-Plus* vorantreiben und unter anderem Teilergebnisse aus dieser Arbeit beinhalten. An beiden Veröffentlichung war der Autor dieser Arbeit als Co-Autor beteiligt. Beide erfolgten auf fachspezifischen Konferenzen, zum einen der ANNSIM 2025 (Annual Modeling and Simulation Conference) [WM25a] und zum anderen der I3M 2025 (International Multidisciplinary Modeling & Simulation Multiconference) [WM25b].

7 Literaturverzeichnis

Alle Internetquellen wurden im Laufe der Bearbeitungszeit abgerufen und zum 17. Juli 2025 auf Verfügbarkeit und Korrektheit überprüft.

Literatur

[Wie24] Thomas Wiedemann. „Simulationsgestützte Analyse der neuen dynamischen Strompreise und abgeleitete Empfehlungen zur Abgabenpolitik“. In: *ASim 2024* (2024).

[Mic25] mdr Michael Houben. *Warum Kraftwerke trotz Dunkelflaute Nicht anspringen*. Jan. 2025. URL: https://www.tagesschau.de/wirtschaft/energie/dunkelflauten-strom-preise-deutschland-102.html.

[Bun24a] Bundesnetzagentur. 2024. URL: https://www.bundesnetzagentur.de/DE/Allgemeines/DieBundesnetzagentur/Insight/Texte/Energiewende/Dunkelflaute.html.

[Zei25] Dein Stromvertrag für das digitale Zeitalter Tibber. Apr. 2025. URL: https://tibber.com/de.

[Bun25a] Bundesnetzagentur. Jan. 2025. URL: https://www.bundesnetzagentur.de/SharedDocs/Pressemitteilungen/DE/2025/20250108_EE.html?nn=694052.

[Möb25] Lukas Möbius. „Interaktive Webanwendung zur simulationsgestützten Analyse und Bewertung von dynamischen Strompreisen und Vergleich alternativer Abgabensysteme“. In: *HTW Dresden, Fak. Informatik/Math (noch unveröffentlicht)* (2025).

[Jus23] Bundesministerium der Justiz. Dez. 2023. URL: https://www.gesetze-im-internet.de/enwg_2005/__41a.html.

[Bun24b] Bundesnetzagentur. Apr. 2024. URL: https://www.bundesnetzagentur.de/DE/Vportal/Energie/PreiseAbschlaege/Tarife-table.html.

[Ene25] EnergyAPI. Apr. 2025. URL: https://api.energy-charts.info/.

[Fra25] Fraunhofer. Apr. 2025. URL: https://www.ise.fraunhofer.de/.

[Bun25b] Statistisches Bundesamt. „Stromerzeugung 2024: 59,4 % aus erneuerbaren Energieträgern“. In: (März 2025). URL: https://www.destatis.de/DE/Presse/Pressemitteilungen/2025/03/PD25_091_43312.html.

[MM89] Friedemann Mattern und Horst Mehl. „Diskrete Simulation - Prinzipien und Probleme der Effizienzsteigerung durch Parallelisierung“. In: *Inform. Spektrum* 12 (1989), S. 198–210. URL: https://api.semanticscholar.org/CorpusID:28041838.

[Pag+88] Bernd Page u. a. *Simulation und moderne programmiersprachen: Modula-2, C, Ada*. Springer Berlin Heidelberg, 1988. DOI: 10.1007/978-3-642-83394-6.

[Cor22a] Microsoft Corporation. *Microsoft SQL Server 2022 Licensing guide*. 2022. URL: https://go.microsoft.com/fwlink/p/?linkid=2215573&clcid=0x407&culture=de-de&country=de.

[Cor22b] Microsoft Corporation. *Microsoft Analysis Services Projects 2022*. Mai 2022. URL: https://marketplace.visualstudio.com/items?itemName=ProBITools.MicrosoftAnalysisServicesModelingProjects2022.

[Mes+17] Abdennacer Ben Messaoud u. a. *A New Strategy of Validities' Computation for Multimodel Approach: Experimental Validation*. Techn. Ber. 2017. URL: www.ijacsa.thesai.org.

[MTK18] Abdennacer Ben Messaoud, Samia Talmoudi und Moufida Ksouri-Lahmari. „Multimodel approach for modelling of nonlinear systems: Validities' optimal computation“. In: *COMPEL - The International Journal for Computation and Mathematics in Electrical and Electronic Engineering* 37 (1 2018), S. 153–175. ISSN: 03321649. DOI: 10.1108/COMPEL-11-2016-0514.

[AF17] Ahmed Adebowale Adeniran und Sami El Ferik. „Modeling and Identification of Nonlinear Systems: A Review of the Multimodel Approach - Part 1“. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 47 (7 Juli 2017), S. 1149–1159. ISSN: 21682232. DOI: 10.1109/TSMC.2016.2560147.

[HN24] Yilin Huang und Igor Nikolic. „Towards a multi-model infrastructure for integrated decision-making in energy transition“. In: *Proceedings of the 23rd International Conference on Modelling and Applied Simulation* (2024). DOI: 10.46354/i3m.2024.mas.002.

[Aja+06] Newsha K Ajami u. a. *Multimodel Combination Techniques for Analysis of Hydrological Simulations: Application to Distributed Model Intercomparison Project Results*. Techn. Ber. 2006, S. 755–768. DOI: https://doi.org/10.1175/JHM519.1. URL: https://journals.ametsoc.org/view/journals/hydr/7/4/jhm519_1.xml.

[GC06] Peter Gluchowski und Peter Chamoni. „Entwicklungslinien und architekturkonzepte des on-line analytical processing“. In: *Analytische Informationssysteme* (2006), S. 143–176. DOI: 10.1007/3-540-33752-0_8.

[SDO17] Object Management Group Standards Development Organization (OMG SDO). „OMG Unified Modeling Language (OMG UML) Version 2.5.1 formal/2017-12-05“. In: *UML 2.5* (Mai 2017). URL: http://www.omg.org/spec/UML/2.5/PDF/.

[Cor22c] Microsoft Corporation. *Key Concepts in MDX (Analysis Services)*. Abgerufen am 14. Juli 2025. 2022. URL: https://learn.microsoft.com/en-us/analysis-services/multidimensional-models/mdx/key-concepts-in-mdx-analysis-services?view=sql-analysis-services-2025.

[Azu25] Microsoft Azure. Juli 2025. URL: https://azure.microsoft.com/de-de/pricing/calculator.

[LO15] Dan Linstedt und Michael Olschimke. *Building a Scalable Data Warehouse with Data Vault 2.0*. Morgan Kaufmann, 2015. ISBN: 9780128025109.

[WM25a] Thomas Wiedemann und Tom Marinovic. „Multimodel validation of simulation scenarios for optimization of the new dynamic electricity prices“. In: *Annual Modeling and Simulation Conference 2025* (2025).

[WM25b] Thomas Wiedemann und Tom Marinovic. „An architecture for distributed validation of multiple simulation models for optimization of the new dynamic electricity prices“. In: *13th International Workshop on Simulation for Energy, Sustainable Development and Environment (SESDE 2025), held within the 22nd International Multidisciplinary Modeling and Simulation Multiconference (I3M 2025) (noch unveröffentlicht)* (2025).

8 Anlagen

Anlage 1 OLTP SQL Befehle

```sql
CREATE TABLE POWER_STORAGE (
    storage_ID INT NOT NULL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    capacity_kWh INT NOT NULL,
    average_charge_rate_kWh_h DECIMAL(12,6) NOT NULL,
    comment VARCHAR(255)
);

CREATE TABLE CONSUMPTION (
    consumption_ID INT NOT NULL PRIMARY KEY,
    consumer_type VARCHAR(255),
    average_consumption_monthly_kWh DECIMAL(12,6) NOT NULL,
    average_consumption_daily_kWh DECIMAL(12,6) NOT NULL,
    comment VARCHAR(255)
);

CREATE TABLE SIM_ALGORITHM (
    alg_ID INT NOT NULL PRIMARY KEY,
    algorithm VARCHAR(255),
    comment VARCHAR(255),
    storage_ID INT,
    FOREIGN KEY (storage_ID) REFERENCES POWER_STORAGE(storage_ID)
);

CREATE TABLE SIMULATION_RESULTS_TEMP (
    sim_ID INT NOT NULL PRIMARY KEY,
    alg_ID INT NOT NULL,
    Sim_date DATETIME NOT NULL,
    Sim_Price DECIMAL(12,6) NOT NULL,
    Sim_Taxes DECIMAL(12,6) NOT NULL,
    FOREIGN KEY (alg_ID) REFERENCES SIM_ALGORITHM(alg_ID)
);

CREATE TABLE POWER_PRICE_HISTORY (
    power_ID INT NOT NULL PRIMARY KEY,
    DatumID DATETIME NOT NULL,
    Price_Brutto DECIMAL(12,6) NOT NULL,
    Price_Netto DECIMAL(12,6) NOT NULL,
    Taxes DECIMAL(12,6) NOT NULL,
    Provider VARCHAR(255),
    comment VARCHAR(255)
);

CREATE TABLE RUN_INSTANCE (
    alg_ID INT NOT NULL,
    consumption_ID INT NOT NULL,
    storage_ID INT NOT NULL,
    Month DATETIME NOT NULL,
    Price_Brutto_Mean DECIMAL(12,6) NOT NULL,
    Price_Netto_Mean DECIMAL(12,6) NOT NULL,
    Price_Min DECIMAL(12,6) NOT NULL,
    Price_Max DECIMAL(12,6) NOT NULL,
    PRIMARY KEY (alg_ID, consumption_ID, storage_ID),
    FOREIGN KEY (alg_ID) REFERENCES SIM_ALGORITHM(alg_ID),
    FOREIGN KEY (consumption_ID) REFERENCES CONSUMPTION(consumption_ID),
    FOREIGN KEY (storage_ID) REFERENCES POWER_STORAGE(storage_ID)
);
```

Anlage 2 Data Warehouse SQL Befehle

```sql
CREATE TABLE TIME (
    TIME_ID INT NOT NULL PRIMARY KEY,
    Year VARCHAR(4),
    Month VARCHAR(20),
    Day VARCHAR(2),
    Hour VARCHAR(2),
    Minute VARCHAR(2)
);

CREATE TABLE PREDICTION (
    PREDICTION_ID INT NOT NULL PRIMARY KEY,
    Date_SIM DATETIME NOT NULL,
    Price_Brutto DECIMAL(12,6) NOT NULL,
    Price_Netto DECIMAL(12,6) NOT NULL,
    Taxes DECIMAL(12,6) NOT NULL,
    SimType_Code INT,
    comment VARCHAR(255)
);

CREATE TABLE POWER_PRICE_HISTORY (
    power_ID INT NOT NULL PRIMARY KEY,
    dateID DATETIME NOT NULL,
    Price_Brutto DECIMAL(12,6) NOT NULL,
    Price_Netto DECIMAL(12,6) NOT NULL,
    Taxes DECIMAL(12,6) NOT NULL,
    Provider VARCHAR(255),
    comment VARCHAR(255)
);

CREATE TABLE CONSUMER (
    CONSUMER_ID INT NOT NULL PRIMARY KEY,
    Type VARCHAR(255) NOT NULL,
    comment VARCHAR(255),
    average_consumption_monthly_kWh DECIMAL(12,6) NOT NULL,
    average_consumption_daily_kWh DECIMAL(12,6) NOT NULL
);

CREATE TABLE SIM_MODEL (
    SIMULATION_ID INT NOT NULL PRIMARY KEY,
    algorithm VARCHAR(255),
    comment VARCHAR(255),
    run_date DATETIME
);

CREATE TABLE SIM_COMPARE (
    TIME_ID INT NOT NULL,
    PREDICTION_ID INT NOT NULL,
    CONSUMER_ID INT NOT NULL,
    SIMULATION_ID INT NOT NULL,
    Price DECIMAL(10,6) NOT NULL,
    Savings DECIMAL(10,6) NOT NULL,
    Validation_score VARCHAR(255),
    PRIMARY KEY (TIME_ID, PREDICTION_ID, CONSUMER_ID, SIMULATION_ID),
    FOREIGN KEY (TIME_ID) REFERENCES TIME(TIME_ID),
    FOREIGN KEY (PREDICTION_ID) REFERENCES PREDICTION(PREDICTION_ID),
    FOREIGN KEY (CONSUMER_ID) REFERENCES CONSUMER(CONSUMER_ID),
    FOREIGN KEY (SIMULATION_ID) REFERENCES SIM_MODEL(SIMULATION_ID)
);
```

Anlage 3 Python Skript API Abruf

```python
import requests
import json
from datetime import datetime, timezone

def fetch_and_export_power_prices(start, end, output_file):
    url = 'https://api.energy-charts.info/price'
    params = {
        'bzn': 'DE-LU',
        'start': start,
        'end': end
    }
    headers = {'accept': 'application/json'}

    try:
        response = requests.get(url, headers=headers, params=params)
        response.raise_for_status()
        data = response.json()

        unix_seconds = data.get("unix_seconds", [])
        readable_times = [datetime.fromtimestamp(ts, tz=timezone.utc).strftime('%Y-%m-%d %H:%M:%S') for ts in unix_seconds]

        transformed_data = {
            "times": readable_times,
            "price": data.get("price"),
            "unit": data.get("unit"),
        }

        with open(output_file, 'w') as file:
            json.dump(transformed_data, file, indent=4)
        print(f"Data successfully exported to {output_file}")

    except requests.exceptions.RequestException as e:
        print(f"API request failed: {e}")
    except Exception as e:
        print(f"An error occurred: {e}")

if __name__ == "__main__":
    start_time = 1704063600  # start UNIX time
    end_time = 1744927200    # end UNIX time
    output_file_name = "power_prices.json"
    fetch_and_export_power_prices(start_time, end_time, output_file_name)
```

Anlage 4 Python Skript Generierung Insert Befehle

```python
import json
import random
from datetime import datetime

PROVIDER = 'https://api.energy-charts.info'
DATE = (datetime.now()).strftime("%Y/%m/%d, %H:%M:%S")
COMMENT = 'Imported on '+ DATE

with open('power_prices.json') as f:
    data = json.load(f)

times = data["times"]
prices = data["price"]

with open('insert_power_data_import.sql', 'w') as sql_file:
    for i, (timestamp, price_netto) in enumerate(zip(times, prices), start=1):
        datum = datetime.strptime(timestamp, "%Y-%m-%d %H:%M:%S").date()

        taxes = round(random.normalvariate(0.23, 0.022), 4)
        price_brutto = round((price_netto/1000) + taxes, 4)

        statement = f"""INSERT INTO POWER_PRICE_HISTORY (
        power_ID, dateID, Price_Brutto, Price_Netto, Taxes, Provider, comment
        ) VALUES (
        {i}, '{datum}', {price_brutto}, {price_netto/1000}, {taxes}, '{PROVIDER}', '{COMMENT}'
        );"""

        sql_file.write(statement)

print("SQL insert statements saved to insert_power_data_import.sql")
```

Anlage 5 DynPlus Generierung Insert Befehle

```python
import pandas as pd
from datetime import datetime

csv_path = 'C:\\Users\\marin\\Documents\\simlog.csv'
output_sql_file = 'simulation_results_inserts.sql'
alg_id = 107  # Static algorithm ID

# --- READ CSV ---
df = pd.read_csv(csv_path, sep=';', encoding='cp1252')
df = df[df['DatumID'].astype(str).str.match(r'^\d{10}$')]

df['Sim_date'] = pd.to_datetime(df['DatumID'].astype(str), format='%Y%m%d%H')

# Clean and convert DynPlusPreis to decimal
df['Sim_Price'] = (
    df['DynPlusPreis']
    .str.replace('€', '', regex=False)
    .str.replace(',', '.', regex=False)
    .astype(float)
)

df['Sim_Taxes'] = 0.0
df['alg_ID'] = alg_id
df['comment'] = df['Text22']

# Generate sim_IDs (e.g., 1 to n)
df['sim_ID'] = range(1, len(df) + 1)

# --- GENERATE INSERT STATEMENTS ---
insert_statements = []

for _, row in df.iterrows():
    insert = f"""INSERT INTO SIMULATION_RESULTS_TEMP (sim_ID, alg_ID, Sim_date, Sim_Price, Sim_Taxes, comment)
VALUES ({row['sim_ID']}, {row['alg_ID']}, '{row['Sim_date'].strftime('%Y-%m-%d %H:%M:%S')}', {row['Sim_Price']:.6f},
{row['Sim_Taxes']:.6f}, '{row['comment']}');"""
    insert_statements.append(insert)

# --- SAVE TO FILE ---
with open(output_sql_file, 'w', encoding='utf-8') as f:
    f.write('\n'.join(insert_statements))

print(f"SQL insert statements written to {output_sql_file}")
```

Anlage 6 Validierung Stored Procedure

```
ALTER PROCEDURE dbo.usp_CompareSimPrices
(
    @BaseAlgID    INT,
    @CompareAlgID INT,
    @StartDate    DATETIME,
    @EndDate      DATETIME
)
AS
BEGIN
    SET NOCOUNT ON;

    -- Cleanup
    DELETE FROM dbo.SIM_COMPARISON_RESULTS
    WHERE Base_Alg_ID    = @BaseAlgID
      AND Compare_Alg_ID = @CompareAlgID
      AND Sim_Date BETWEEN @StartDate AND @EndDate;

    -- Start <= End
    IF @StartDate > @EndDate
    BEGIN
        RAISERROR(
          'Error Startdate before Enddate.',
          16,1
        );
        RETURN;
    END

    INSERT INTO dbo.SIM_COMPARISON_RESULTS
    (
        Sim_Date,
        Base_Alg_ID,
        Compare_Alg_ID,
        Base_Sim_Price,
        Compare_Sim_Price,
        Percent_Difference
    )
    SELECT
        b.Sim_Date,
        b.alg_ID     AS Base_Alg_ID,
        c.alg_ID     AS Compare_Alg_ID,
        b.Sim_Price  AS Base_Sim_Price,
        c.Sim_Price  AS Compare_Sim_Price,
        CASE WHEN b.Sim_Price = 0 THEN NULL
             ELSE ((c.Sim_Price - b.Sim_Price) / b.Sim_Price) * 100
        END AS Percent_Difference
    FROM
        dbo.SIMULATION_RESULTS_TEMP AS b
        INNER JOIN dbo.SIMULATION_RESULTS_TEMP AS c
            ON b.Sim_Date = c.Sim_Date
    WHERE
        b.alg_ID     = @BaseAlgID
        AND c.alg_ID = @CompareAlgID
        AND b.Sim_Date BETWEEN @StartDate AND @EndDate
        AND b.Sim_Price IS NOT NULL
        AND c.Sim_Price IS NOT NULL;

    DECLARE @InsertedRows INT = @@ROWCOUNT;
    PRINT CONCAT('Inserted ', @InsertedRows, ' new Validationscores.');
END
GO
```

Anlage 7 Aufruf Validierungsprozedur

```
EXEC dbo.usp_CompareSimPrices
    @BaseAlgID    = 107,
    @CompareAlgID = 108,
    @StartDate    = '2024-01-01 00:00:00',
    @EndDate      = '2024-06-01 00:00:00';

SELECT * FROM SIM_COMPARISON_RESULTS WHERE NOT Percent_Difference = 0

SELECT
    alg_ID,
    Sim_Date,
    COUNT(*) AS CountPerKey
FROM dbo.SIMULATION_RESULTS_TEMP
WHERE alg_ID IN (107,108)
  AND Sim_Date BETWEEN '2024-01-01 00:00:00' AND '2024-06-01 00:00:00'
GROUP BY alg_ID, Sim_Date
HAVING COUNT(*) > 1;
```

9 Selbstständigkeitserklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe. Ich reiche sie erstmals als Prüfungsleistung ein. Mir ist bekannt, dass ein Betrugsversuch mit der Note „nicht ausreichend“ (5.0) geahndet wird und im Wiederholungsfall zum Ausschluss von der Erbringung weiterer Prüfungsleistungen führen kann.

Name: Marinovic

Vorname: Tom

Matrikelnummer: 55156

Dresden, den 18.07.2025

Unterschrift