International Journal of
Geo-Information

Article

Domain-Specific Language for Land Administration System Transactions

Đorđe Pržulj, Igor Dejanović, Miroslav Stefanović *, Teodora Lolić and Srđan Sladojević

Faculty of Technical Sciences, University of Novi Sad, 21000 Novi Sad, Serbia; przulj@uns.ac.rs (Đ.P.); igord@uns.ac.rs (I.D.); teodora.lolic@uns.ac.rs (T.L.); sladojevic@uns.ac.rs (S.S.)
* Correspondence: mstef@uns.ac.rs

Abstract: The Land Administration System (LAS) records real estates, owners, and rights information. Changes that take place in the real world are recorded as transactions in LAS. This paper discusses various data-integrity constraints that have to be taken into account so that LAS data will be correct and consistent after the execution of LAS transactions. Those transactions are executed by system users, typically through some graphical user interface (GUI) applications. Domain-specific languages (DSLs) provide the possibility for domain experts to write statements that can be interpreted and executed on respective software systems. In the case of LAS, DSL for LAS transactions could enable land administration experts to write statements that would execute transactions and keep LAS data up to date with real world changes. Two types of LAS transactions are considered: legal transactions, which result in ownership changes, and survey transactions, which change the real estate geometry data. In this paper, a possible DSL solution for transactions in the LAS domain is proposed. A system architecture that could enable the efficient writing, validation, verification, execution, and storage of DSL statements is also proposed. A possible DSL for LAS transaction implementation is presented, and examples of legal and survey transactions are explained. The advantages and possible challenges of the proposed solution's implementation are also discussed in this paper.

Keywords: land administration; land administration transactions; domain-specific language

check for updates

Citation: Pržulj, Đ.; Dejanović, I.; Stefanović, M.; Lolić, T.; Sladojević, S. Domain-Specific Language for Land Administration System Transactions. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 425. https://doi.org/10.3390/ijgi11080425

Academic Editors: Dev Raj Paudyal and Wolfgang Kainz

Received: 29 June 2022
Accepted: 26 July 2022
Published: 27 July 2022

1. Introduction

Land Administration System (LAS) is a system that registers real estates, their ownership, value, and use. LAS describes real property with physical, spatial, topographic, and thematic attributes. LAS data can informally be divided into two subsets: data describing rights and restrictions on real properties (legal data) and data describing the geometry of real properties (survey data). To support such data diversity, LAS contains two complementary subsystems: land registry and cadaster.

Land registry is an official record of rights related to land or of deeds concerning changes in the legal situation of defined units of land. It answers the questions of who owns particular property and what legal document that ownership is based on [1].

Cadaster represents a public inventory of data regarding properties within a certain county or district. Data in cadaster are based on a survey of boundaries. It represents a register of spatial data used for describing properties and answers questions about the location of a certain property [1]. Data that may appear in a cadaster include geometric data (coordinates, maps) [2–4], property addresses [5,6], land use [7,8], real property information, the nature and duration of the tenure, details about the construction of buildings and apartments [9–12], and land taxation values [13,14]. Furthermore, cadastral data models help to gather and supply relevant information for disaster management [15], sustainable development [16], food security [17,18], post-conflict administration [19], and spatial planning [20].

In 'Cadaster 2014' [21] a vision for a cadaster system 'without paper and pencil' is presented. As a result of this research, an international initiative for common data-model development was initiated. The result was the creation of the ISO 19152:2012 Geographic information—Land Administration Domain Model (LADM). The LADM is a conceptual data model that is widely adopted as the descriptive standard for land administration [22]. Many country profiles and LADM application analyses have been published to date. After 'Cadaster 2014', the focus of academia is on 'Cadaster 2034' and what we could expect from LAS in years to come. Research is focused in two directions. The first direction is based on the new technological achievements that could bring survey-accuracy, object-oriented design, 3D/4D arrangements, real-time information, global linkages, and organic characteristics as the main characteristics of future cadasters [23]. The second direction is related to prioritising important social, legal, and environmental agendas, such as securing basic rights, providing for equity, fairness, transparency, accountability and the rule of law, and government decentralisation [24].

One of LAS's tasks is to keep the data up to date with the changes in the real world related to the land administration domain. Those changes include changes of ownership (buying/selling, inheritance), the establishment of rights (such as mortgage), and the division or merging of parcels [25]. Managing LAS data is conducted by executing transactions that update the land registry and cadastral data. In the digital era, LAS manages legal data describing owners, real estates, and ownership rights and restrictions, such as the owner's name, parcel area, or the ownership share. On the other hand, LAS also has to manage the survey data that describe real estate shape and position by vector graphics using an appropriate coordinate reference system. Some LASs may contain additional thematic data that describe land usage or similar data describing the purpose of spatial features on the terrain [26].

There are differences between the LASs in different countries. Land registration systems may be based on deeds or titles. An LAS may be implemented as a dual system, with a separate land book (title holders on real property) and land cadaster (real property) or as a unified system with one register (in some countries it is called real estate cadaster). In both cases, the problem of the integration of the land registry and cadastral data remains an important issue, due to the fact that these two subsystems are loosely coupled. Moreover, both subsystems have their own business rules and transactions [27].

After the LAS initial state is established by collecting and arranging real-world data, the main activity in LAS management is to keep those data up to date by executing transactions that reflect real-world changes, such as ownership rights changes, new building registration, owner address updates, and so on. These transactions have to maintain data correctness and consistency, which may be violated especially if a transaction spans over both the land registry and cadaster. The analysis presented in [28] shows an example of the significant inconsistency between data in the land register and cadaster.

ISO 19157:2013 Geographic information—Data quality defines six data quality elements: completeness, thematic accuracy, logical consistency, temporal quality, positional accuracy, and usability [29]. The data quality should be evaluated on the base of metadata stored together with legal and survey data in LAS. ISO 19115-1:2014 Geographic information—Metadata—Part 1: Fundamentals and ISO 19115-2:2019 Geographic information—Metadata—Part 2: Extensions for acquisition and processing define schema required for describing geographic information and services by means of metadata [30,31]. Data quality as well as semantics and terminology still represent the major challenge in contemporary LASs.

Today, in an era of powerful computer systems, LAS efficiently stores, updates, and manages data in digital format. An LAS data update is typically conducted by LAS employees through graphical user interface (GUI) applications. Those applications are developed using various general-purpose languages (GPL). Using GUI applications requires basic computer usage skills but no programming knowledge.

Domain-specific languages (DSLs) are languages developed, as its name suggests, to be used in a specific domain, contrary to GPL. GPLs can be used to build general computer software and typically have great possibilities and flexibility, while DSLs are typically easy to use and write statements, so users must have appropriate domain knowledge and basic programming skills. DSL statements are then translated or interpreted into some GPL statements.

Defining and using DSLs is a kind of model-driven software development. The first step in development is the creation of formal, tool-processable representations of specific aspects of software systems. These representations (DSL statements) can be directly interpreted by interpreters or transformed into GPL statements by code generators [32].

The motivation for this research was to develop a DSL for the land administration domain so the user can write a statement that will execute as a transaction in the LAS. The research is based on the use case of land administration transactions in Serbia. The Serbian LAS, as well as the LASs of other Western Balkan countries, is based on the Austro-Hungarian land registry and cadaster. There are multiple DSL implementation benefits. DSL statements can be used to execute various complex operations that are difficult to execute using standard GUI applications. The presentation and reporting of DSL statements can be conducted in the same format as they are written. DSL statements can be validated and verified by an appropriate integrated development environment, so the end user efficiency is improved. Those statements are understandable to the user that has no programming knowledge or skills. DSL statements could be translated into multiple-target languages [33]. This paper proposes DSL for LAS transactions that would enable land administration employees to write statements that would be executed to keep LAS data up to date with real-world changes.

DSL for LAS transaction statement persistence could be implemented in centralized systems, such as database systems, or in a distributed transaction ledger, such as blockchain. Since blockchain technology already provides smart contracts as a mechanism to execute transactions generally, it will be compared with the solution of DSL for LAS transaction statements stored in an arbitrary blockchain.

This paper is organized as follows: Section 2 gives some theoretical background of the fields of LAS and DSL. Various types of LAS transactions are discussed in Section 3. In Section 4, the system architecture for DSL for LAS transactions is discussed. The proposed solution is presented, with examples of LAS transaction statements, in Section 5. The discussion and future work guidelines are given in Section 6.

2. Related Work

In 2012, the International Standardisation Organization (ISO) published ISO-19152:2012 Geographic information—Land Administration Domain Model (LADM), which defines a reference domain model for covering basic information-related components of land administration [22]. LADM does not provide a universal solution suitable for any specific LAS. Instead, it is recommended as a reference for the creation of domain models suitable for a specific country profile. In the domain of geographic information, a profile is a "set of one or more base standards or subsets of base standards, and, where applicable, the identification of chosen clauses, classes, options, and parameters of those base standards that are necessary for accomplishing a particular function" [34]. As a result, many country profiles based on LADM have been published, such as [35–38]. According to [39] LADM is mainly the focus of academia. The main subjects of research are the creation of new LADM country profiles and how current systems match LADM. In some cases, LADM expanded to add additional layers of data, such as land use [26], as well as to specify additional constraints that exist in LASs, especially regarding the internal consistency of data between the land registry and cadaster [27]. Transactions related to LASs are outside the scope of LADM because the transactions being significantly country-specific. The term 'transaction' is used here to describe the transformation of the state, which has properties of atomicity, consistency, isolation, and durability. In 2018, a New Working Item Proposal for Edition II of LADM was proposed that included multiple extensions, including transactions and the application of blockchain technologies [36,40].

In general, transactions in LAS could consist not only of changing the ownership of real estate, establishing rights related to real estate such as mortgages or development and easement rights, and the merging and splitting of real estates, but could also be related to improving the quality of data stored in LAS [25]. As mentioned, these procedures differ a lot from country to country and have different actors involved. For example in the case of Turkey, the General Directorate of Land Registry and Cadaster performs 25 different cadastral and land registry transactions, requiring different documents from different external institutions [41]. In the case of Slovenia, the typical transfer of agricultural land involves at least five actors: the seller, buyer, notary, administration office, and LAS and includes the execution of at least 12 steps/activities between actors [42]. The process of real estate transactions in Sweden also involves several actors: seller, buyer, real estate agents, land administration office, seller, and buyer's banks and involves 34 steps [43]. Even with just a few examples, it is clear that the registration of real estate transactions could be a rather complex process, and that it can differ significantly from country to country or, to be more precise, from one legal jurisdiction to another.

ISO Technical Specification (ISO/TS) 19103:2015 (ISO 19103: Geographic information—Conceptual schema language, 2015) defines Unified Modeling Language (UML) as a conceptual schema language for the specification of geographic information [44]. So, in most cases, UML is used to describe the current process of real estate transactions, usually through class diagrams, use cases, and sequence diagrams. Based on those UML diagrams, contemporary LAS are usually developed as GUI-based applications, with dedicated components for supporting the manipulation of spatial data. This is conducted in some GPLs created for developing software solutions for a variety of different domains. UML and GPL are the main tools for creating software solutions used by IT professionals. Although the processes of land administration transactions in specific legal jurisdictions are different, the actors and activities involved in those processes are similar and could justify the creation of a DSL for LAS transactions.

A DSL represents the programming or specification language that is developed to solve problems in a specific domain. DSLs provide notations and constructs for application in a specific domain, and, as such, DSLs are more expressive and easier to use than GPL. Moreover, compared to GPL, DSL opens up application domains to a larger group of software developers [45] as well as domain experts. By using DSL, domain experts can be more involved in developing a solution to a problem, because they can more easily understand and validate code and understand what impact a specific change will have [46]. To go a step further, unlike GPL, DSL is intended for the end user, who does not need to be a software developer; so, the user could perform simple programming tasks, using some macro or scripting language [47].

Some of the main benefits of DSL are:

- DSL makes it possible to express transactions at the level of abstraction of the problem domain and in a way makes it possible for domain experts to understand, validate, modify, and in some cases develop DSL programs;
- DSL is self-documented to a large extent for reasons mentioned in the previous statement;
- DSL can improve productivity, maintainability, reliability, and portability;
- DSL incorporates knowledge about a specific domain of application, in that way preserves it and makes it available for future use;
- DSL makes it possible to perform validation and optimization at the domain level [47].

A DSL can be classified by three dimensions: appearance, origin, and implementation [45]. Appearance classifies DSL as one of textual, graphical, tabular, or symbolic. According to the origin, a DSL could be internal, developed from existing GPL or DSL, or external, relying on its own infrastructure. Regarding implementation, DSLs are classified as well-known executions, DSLs that serve as input to application generators; non-executable DSLs, which are, nonetheless, useful as input to application generators; and DSLs not designed to be executed [46].

To the best knowledge of the authors, apart from several papers related to modeling processes in landscapes [48–50] and land use [51,52], which do not closely related to the research presented in this paper, no research has been conducted on developing DSL in the field of LASs or, more specifically, for managing LAS transactions. By proposing DSL for LAS transactions, a new level of abstraction, statement validation, and possibility of interpretation of various platforms are introduced. Domain experts could perform LAS transactions more easily by writing DSL statements.

3. Land Administration System Transactions

All LAS data may be divided into two main datasets: legal and survey. The different natures of those datasets make transaction execution in the integrated LAS a bit challenging. One of the important tasks of LAS is to keep legal and survey data in a consistent and correct state. Some LAS data, such as a parcel area for example, are recorded in both datasets. It is recorded as a numeric value in the legal dataset and can also be calculated from polygon vertex coordinates from the survey dataset. This redundancy is inherited from legacy LASs that have used analog cadastral maps. Polygon area calculation was not automated and was not possible immediately, so the calculated area was recorded in a legal dataset. This way of recording the same data twice in a different datasets creates the possibility for data inconsistency to occur [53,54].

In automated LASs, many inconsistencies between legal and survey data collected from analog sources can be detected by software tools. Legal data are stored in digital form, survey data (analog maps) are digitized, and spatial features are recorded as vector graphic data. A prerequisite for the execution of LAS transactions in an appropriate way by following business rules and constraints is that initial datasets are in a consistent state.

Some LAS transactions are executed in a way that requires several legal and survey CRUD (create, read, update, delete) operations to be executed to finish the whole process. One example of such a transaction is parcel division. The use case scenario for this transaction is as follows:

1. The user selects a parcel from the legal dataset that should be divided;
2. The system loads the parcel survey data and shows it on a digital map;
3. The user divides a parcel using the input data that describe the division line;
4. The system generates labels and calculates the areas of newly created parcels that are the result of division and commits the transaction in the survey dataset;
5. Based on newly created parcels in the survey dataset, the system inserts the new parcel data into the legal dataset, including labels, areas, etc., and commits the transaction;
6. The system deactivates the original parcel that has been divided and commits a transaction in the legal dataset.

In the previous example, in step 2, the legal data are the input for the survey data transaction, and later, in step 4, the survey data are the input for the legal data transaction.

In this paper, changes that include only legal data are referred to as legal transactions. Changes that include only survey data are referred to as survey transactions. In the case that both legal and survey data are being changed, those transactions are referred to as composite transactions.

In Figure 1, the authors are present a UML class diagram for LAS transaction key abstractions from the case study of Serbia.

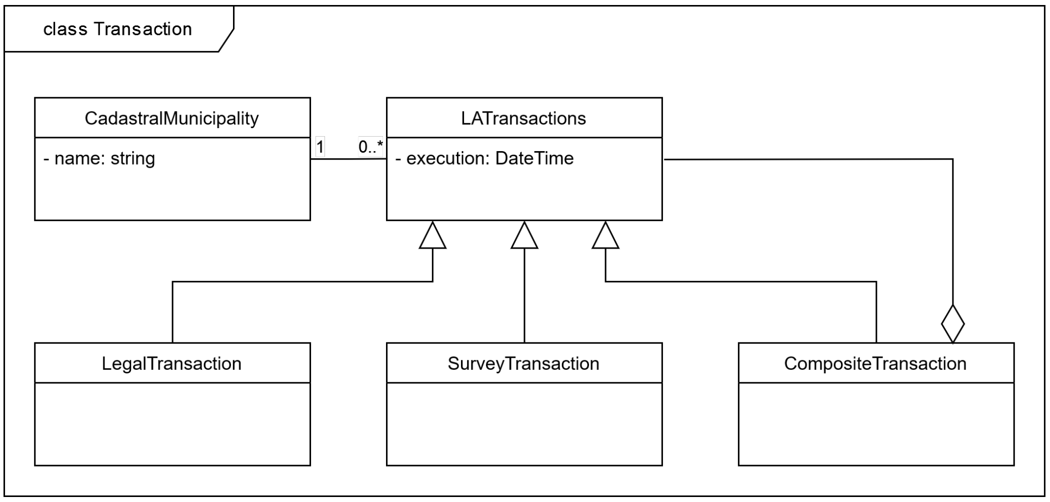


Figure 1. LAS transactions' key abstractions.

The class CadastralMunicipality represents the cadastral spatial units that are used to divide the territory into smaller, more easily manageable administrative parts. Changes that take place on the ground are registered within the cadastral municipality. The approach to register one transaction in one cadastral municipality is inherited from the legacy cadastral systems and may cause complications in situations when parcels from more different cadastral municipalities are involved. This problem is usually solved such that more separate transactions are registered, one for each cadastral municipality involved.

One of the requirements for LASs is to keep track of all transactions that took place and, if required by legal authorities, revert some of them and restore the cadastral data in the state prior to the execution of the transaction. A challenge in these situations may be that the data that should be restored are subsequently changed. In these situations, a typical "undo" approach cannot be conducted.

Transactions in LAS may be atomic or composite. To model its structure and execution, a Composite design pattern is used [55]. This pattern proposes an abstract class Component that has an operation common to its specializations: Leaf for atomic instances and Composite for composite ones. In the case of LAS transactions, the abstract class is LATransaction, and its abstract operation is named execute(); the two atomic classes are LegalTransaction and SurveyTransaction, and the composite one is CompositeTransaction class. By applying a composite design pattern, it is possible to register and execute composite transactions, such as one described in the use case scenario above, in the same way as for atomic transactions.

3.1. Legal Transactions

Typical legal transactions are those that transfer the ownership of real estates from one party to another or add or remove restrictions such as mortgages, changes of land usage, and so on. They may be very simple, such as owner address changes, but they can also be more complex, for example, involving several owners selling their shares of ownership to several different buyers in a different share distribution. These transactions may be a consequence of purchase, gift, inheritance, etc.

3.1.1. Domain Model for Legal Transactions

The LAS legal data domain model is shown in Figure 2. There are a few classes that represent key abstractions.

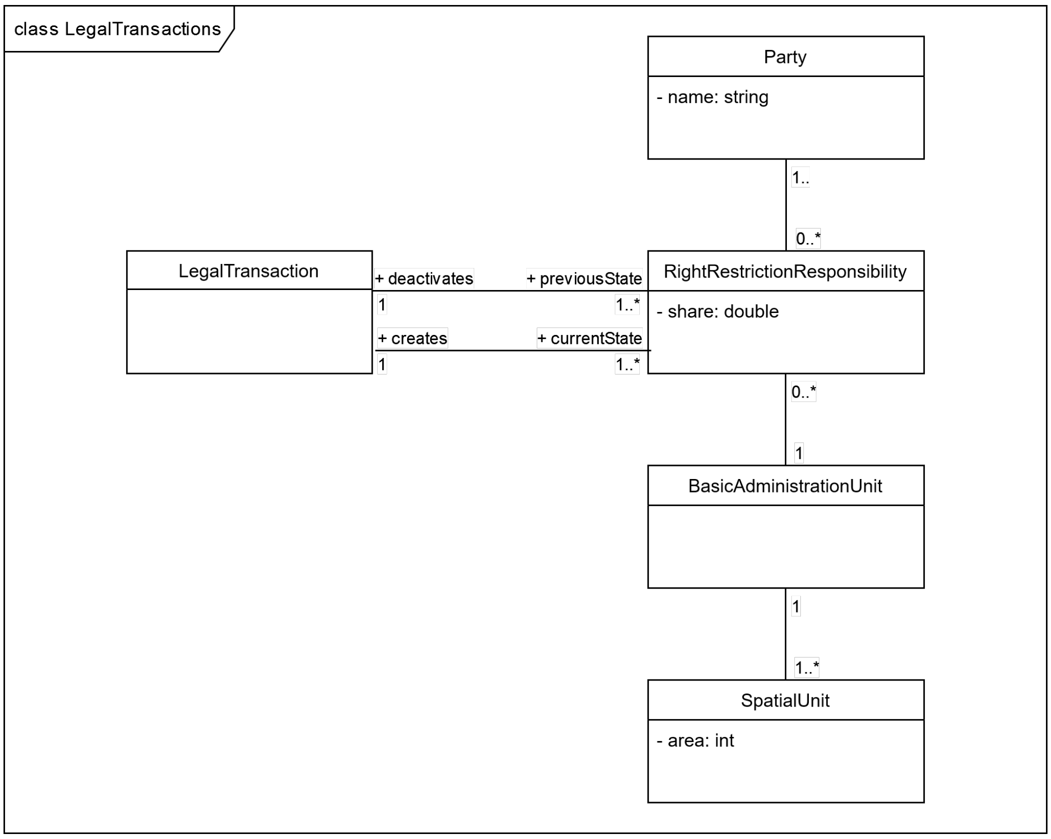


Figure 2. Legal transactions' key abstractions.

The class Party represents owners that could be persons or various types of organizations (companies, institutions, etc.). The class SpatialUnit represents real estate that may be a land parcel, building, or part of the building (apartment, garage, etc.). The class RightRestrictionResponsibility represents the ownership restriction or responsibility of a party on a spatial unit. Since parties may share relationships on the same spatial unit, an attribute share is defined in the class. The model also contains the class BasicAdministrationUnit, which is common in various cadastral systems. The basic administration unit represents a collection of rights, responsibilities, and restrictions with the same ownership relationships. For example, if party p1 has an exclusive right to spatial units su1 and su2 (share 1), these two rights would be grouped in one basic administration unit. If parties p1 and p2 share rights to su3 and su4 in the shares of 0.7 and 0.3, respectively, these rights would be recorded in another basic administration unit. If the same parties, p1 and p2, share half of the rights to the spatial unit su5 each, it would be recorded in another land administration unit.

The concept of a basic administration unit was useful in the legacy cadastral systems to group the collection of rights, responsibilities, and restrictions with the same ownership relationships. Grouped data with the same ownership relationship from the cadastral database can be extracted by executing an appropriate database query on the rights data. It is worth mentioning that, in some cases, digital data might not have the same legal force as the analog data that are still being stored in many national cadasters. Keeping the concept of a basic administration unit in the domain model makes LAS transaction execution a bit more complex since its versioning has to be conducted, too.

The execution of a legal transaction that transfers rights will deactivate all the rights that will be marked as the previous state and create new rights that will present the current state, which will be in effect after the transaction is completed. It is important to note that every right record will have information about which transaction created it and made its current state and which one made it a previous state if any.

3.1.2. Constraints in Legal Transactions

Legal transactions have to conform to constraints and regulations that are defined in the land administration domain. Below are some of them.

- The original owners must possess an ownership share of the real estate that is to be transferred.
- The ownership share that is to be transferred from the original owners to the new owners must be the same.
- The original owners' ownership will be deactivated but must be saved so it can eventually be restored.
- If rights that are to be transferred belong to the basic administration unit that is involved in another transaction that is still not completed, the current transaction execution has to be postponed.

3.2. Survey Transactions

Survey transactions deal with spatial unit geometry data. In two dimensional space, parcel geometry is represented by polygons. Typically, one survey transaction affects a set of parcels that form one connected homogeneous area. In this case, the survey transaction has one outer boundary which may be represented by a closed polyline. There are also cases when parcels affected by a survey transaction are not all connected, so the survey transaction has more outer and inner boundaries. So, here we will introduce the term "transaction boundaries", which represents one or more closed polyline that represents all transactions' outer and inner boundaries.

3.2.1. Domain Model for Survey Transactions

The domain model representing the LAS survey transactions is represented in Figure 3. An important abstraction introduced in this class diagram is the class Geoaggregate. It represents a collection of geoelements that describe the geometry of a parcel. It typically contains polygons, line strings, and individual points depicting parcel features. Polygons represent areas, line strings represent linear features such as paths or retaining walls, while points represent features such as posts, wells, or other features with dimensions not significant on a given map scale.

The class Theme represents any thematic context which a geoelement can be associated with. In the case of polygons, it can be land usage, in the case of line segments it can be a type of fence, while in the case of points, it can be the purpose of a given point feature, such as a post, well, or traffic light, for example.

The class LATransaction creates one or more geoaggregate as a current state and deactivates one or more geoaggregate that represents the previous state of the same territory. Similarly, as rights in the legal transaction domain model, every geoaggregate must contain information about which transaction it has been created by, and which one it has been deactivated by, if any.

A more detailed description of the domain model representing cadastral map data abstractions and their relationships with cadastral legal concepts is available in [26].

3.2.2. Constraints in Survey Transactions

The survey transaction area is covered by geoaggregates created by transaction execution. The topological relationships of these geoaggregates must be correct, so there is no overlapping of geoaggregates or the existence of areas that are not covered by a geoaggregate. This condition will also guarantee that the area of the current state is equal to the area of the previous state. Topological relationships of geoaggregate polygons representing parts of a parcel should also be correct in the same way.

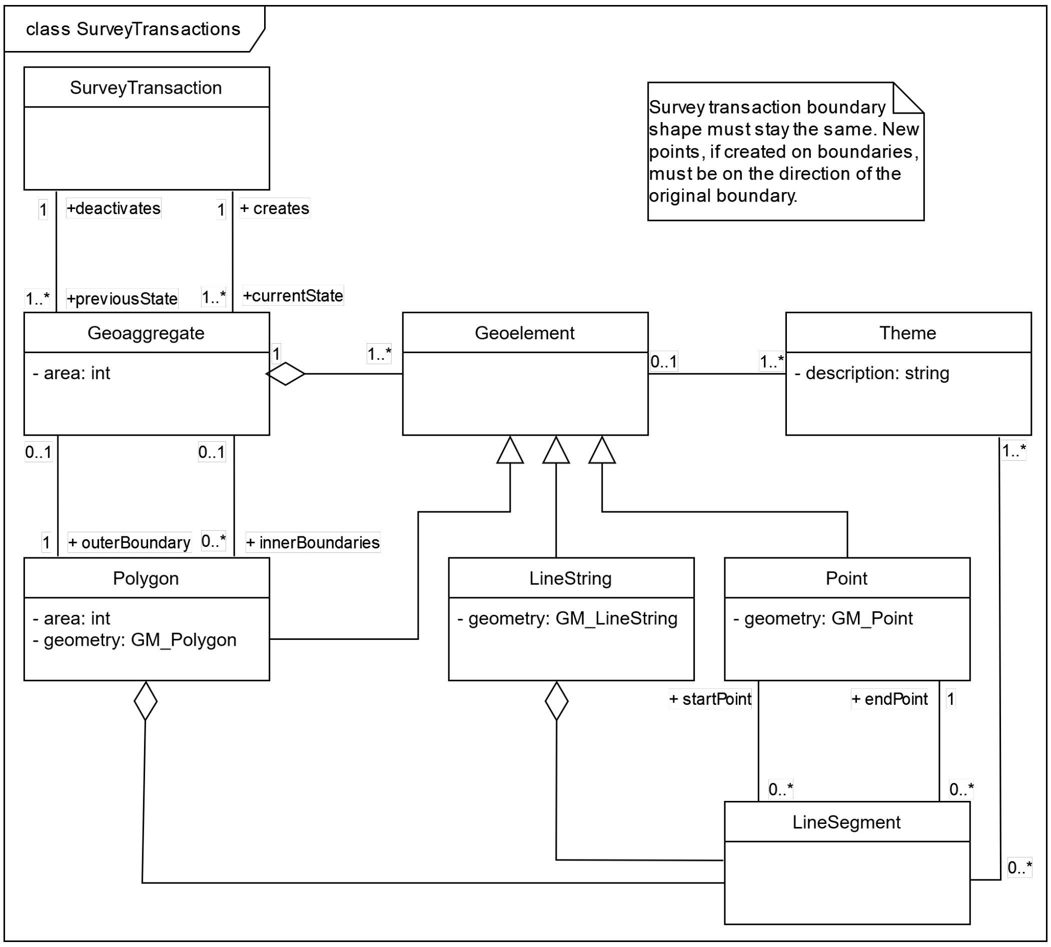


Figure 3. Survey transactions' domain model.

To ensure that the coverage of a survey transaction stays the same, transaction boundaries must cover the same path, not necessarily with the same number of vertices. An example of simple parcel division depicting this rule is given in Figure 4.

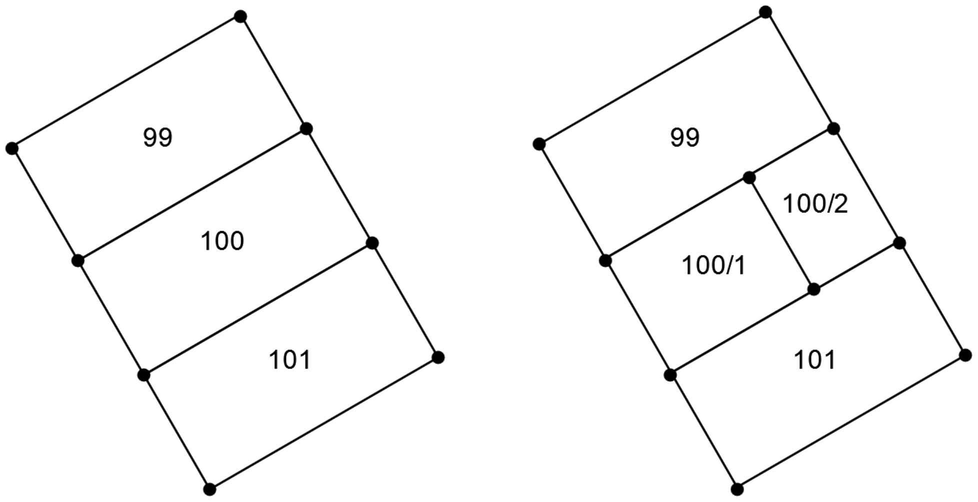


Figure 4. Simple parcel division.

A simple survey transaction that divides a parcel labeled 100, will be considered as an example. Neighboring parcels labeled 99 and 101 are not included in the transactions' previous state. Typically for some cadastral systems, newly created parcels will inherit the original parcels' number and obtain subsequent parcel sub-numbers. According to this rule, newly created parcels will be labeled 100/1 and 100/2. There are also cadastral systems that label newly created parcels with new numbers.

In this example, it can be seen that the geometry of neighboring parcels labeled 99 and 101 is affected by this transaction, even though they were not part of the previous state. After the execution of the transaction, both neighboring parcels now have five instead of the original four vertices. To ensure the shape of affected neighboring parcels is unchanged, every new vertex must be collinear with the endpoints of the divided line.

Another validation rule is that newly created parcels must have a unique label (number and sub-number combination) in the cadastral municipality. These numbers can be generated by software or defined by the user executing the transaction. In the latter case, the software has to check if a user duplicated the parcel label for a given cadastral municipality.

3.3. Composite Transactions

As previously mentioned, changes that include both legal and survey data are referred to as composite transactions. The activity diagram showing the full execution flow of the simple parcel division is presented in Figure 5.

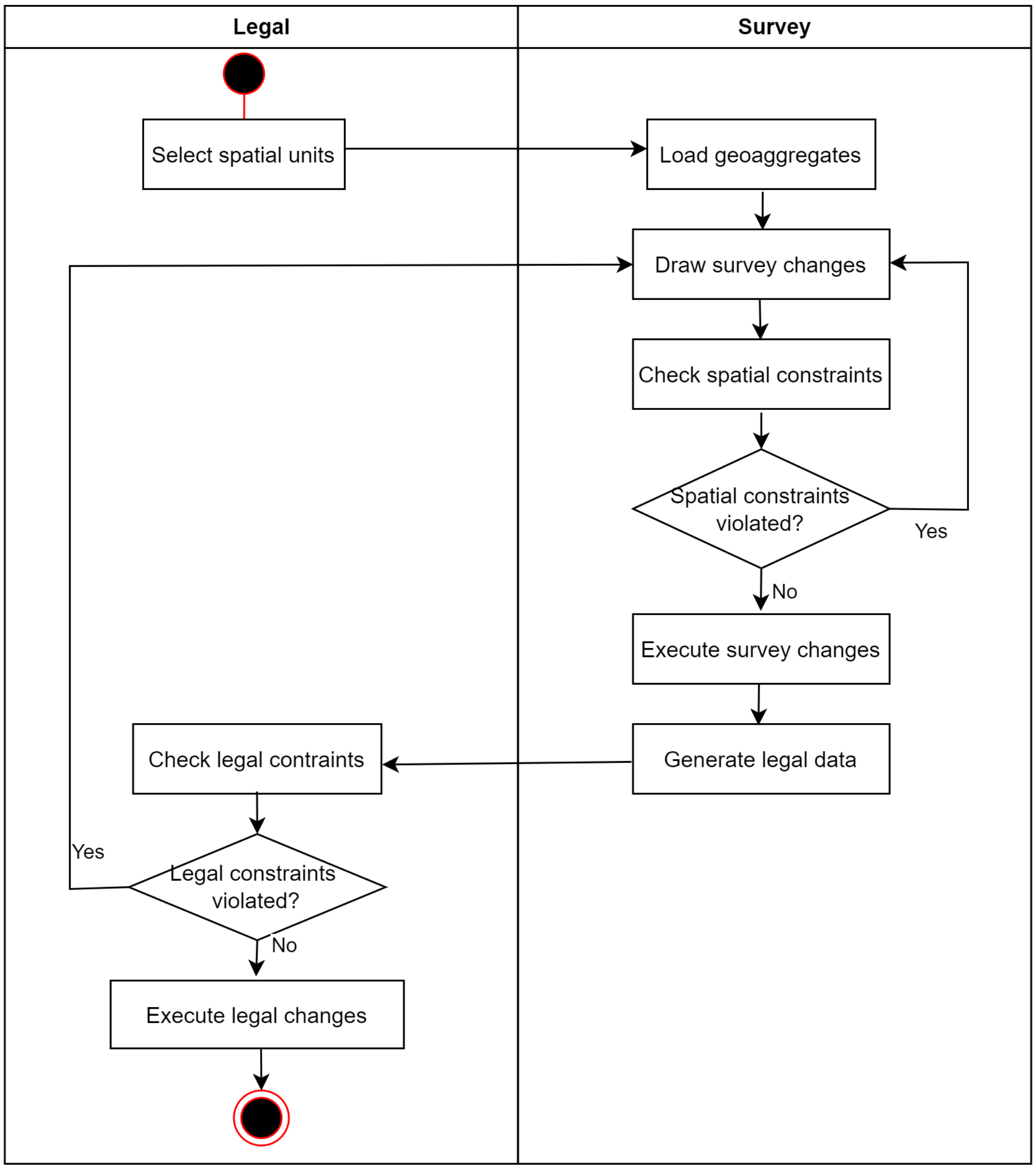


Figure 5. Simple parcel division activity diagram.

From the activity diagram, it can be seen that survey data changes are inputted for legal data changes. Examples of survey data used for legal data generation include parcel labels, parcels and part of parcel areas, and land usage data.

While legal data typically record areas as whole-number values, survey data polygon areas calculated according to their vertex coordinates have decimal number values. Rounding decimal polygon area values can result in a difference between the total area of the previous state and the current state in the legal dataset. Presuming that the polygon representing parcel 100 from the Figure 4 example has an area of 200.6 square meters, parcel 100 in the legal register will have an area rounded to 201 square meters. If we say that polygons representing parcels 100/1 and 100/2 have areas 120.4 and 80.2, respectively, their areas will be rounded to 120 and 80, so, after the transaction, the current state will be one square meter less than the previous state. This problem in the legal changes procedure resolves by leveling areas before and after the transaction. Generally, there can be many parcels affected by the transaction and the area difference for the leveling will be distributed proportionally to the parcel area affected by the transaction.

A comprehensive analysis of a property subdivision use case in Sweden, with the structural analysis conducted using LADM, is described and discussed in [56].

Legal and Survey Data Consistency

Legal and survey data consistency is crucial for LAS data correctness. Some of the constraints that should be checked are:

- parcels' and buildings' labeling compatibility;
- area compatibility;
- thematic data compatibility.

Parcels in LASs are identified by their labels. Typically the label consists of the parcel number or sub-number, if the parcel originates from the previous parcel division. If the legal data transactions are not executed in synchronization with survey data transactions, the LAS will enter an inconsistent state. On the legal data side, there will be parcels that exist without their corresponding geoaggregate in the survey data. Moreover, geoaggregates will appear in the survey data that do not have corresponding parcels in the legal data.

Area compatibility means that the area of the parcel recorded in the legal dataset has to be equal to the area of the corresponding geoaggregate in the survey dataset. The same stands for parts of parcels and their corresponding polygons.

If LAS has a thematic data component, then the mapping that defines which thematic features from the survey dataset correspond to the legal dataset attributes must be defined. For example, various topographic symbols on cadastral maps represent corresponding land usage that is recorded in the legal dataset.

4. System Architecture

As described earlier, LAS transactions can be complex, and one transaction can span over a legal and survey dataset. The execution of these transactions is typically conducted by LAS employees, who use some GUI-based software in order to execute a set of tasks that compose the transaction.

By developing a DSL for LAS transactions, we would provide a language that empowers LAS employees to execute LAS transactions by writing DSL statements. These statements would be interpreted by software written in some GPL. There could be various DSLs for LAS transaction statement interpreters, each of them adapted for the underlying GPL written software. These statements would be easy to store, search, or edit. We can say that DSL for LAS transactions would add a new level of abstraction to LAS transactions execution.

DSL for LAS transactions can also provide the possibility to execute statements chronologically from the LAS zero-state. This would enable users to reconstruct the LAS state at any point in time.

4.1. Proposed System Architecture

From the point of implementation support for DSL, we can identify two main architectures:

- architecture based on compilers;
- architecture based on interpreters [32].

In the first case, DSL statements are being translated into another language that already has defined semantics and a compiler. Architecture based on interpreters provides DSL statement interpretation without translating statements into another language.

Since the DSL for LAS transactions would be used to execute LAS transactions, and there are no languages that already have defined semantics for this purpose, architecture based on interpreters is obviously the right choice.

A possible software system architecture that would provide functionality for writing and executing DSL statements for LAS transactions is presented in Figure 6.

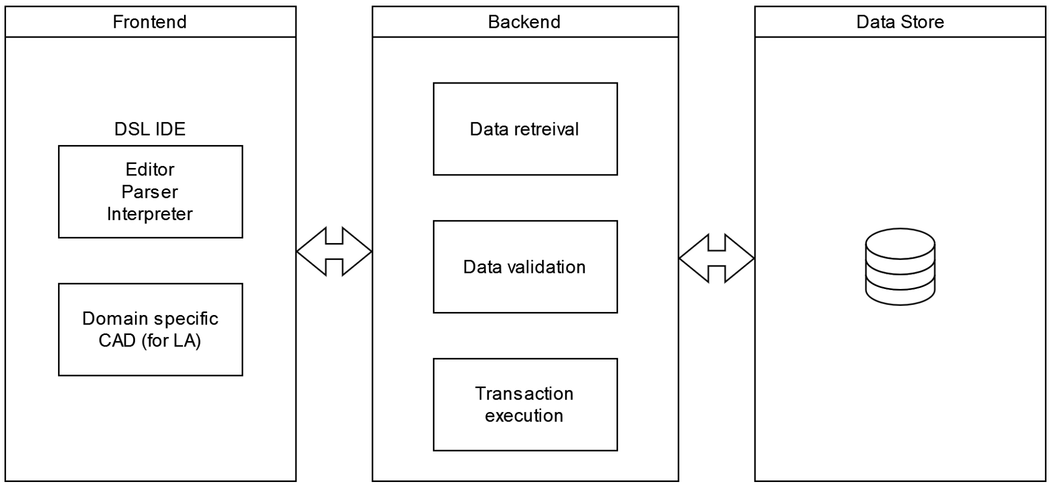


Figure 6. Possible system architecture

4.1.1. Front-End Tier

To write valid DSL statements effectively, the user should be provided with an environment that helps in the process of expression writing, such as syntax highlighting, code completion, etc. After statements are completed, the front-end component should parse the code to verify if the syntax and semantics are correct.

To populate parts of a statement with valid values, the front-end tier could need services provided by the back-end API. For example, to populate the statement that transfers ownership from the seller to the buyer, when writing seller and buyer identifiers in the editor, the front-end component could call the back-end API to check if owners with the given identifiers exist in the database. Furthermore, to be more user-friendly, the front-end could provide the user with an option to search owners by name instead of specifying the identifier. In that case, the front-end would pass search criteria to reach the owners by name, while the back-end API would provide a service to return the owners' data that satisfy the specified request.

After the statements are written, the front-end should employ the parser to check the syntax and semantics of the DSL statements. After the statement's validation, it can be executed. Execution of the statement will be conducted by executing appropriate native language statements. Typically, this will end up as calls to APIs provided by the back-end services.

Another component that is necessary for survey data manipulation is a tool that would provide a user-friendly environment for cadastral map updates. This could be a purposely developed GUI tool that enables land administration users to update survey data. The tool should guide users to produce a valid final result. Different topological analyses, geometry construction methods, validation procedures, and other business rule enforcement mechanisms may be implemented.

Data produced in this way will be referenced from DSL for LAS transaction statements to execute changes to LAS data. This approach will enable users to deal with survey data in a user friendly way, using a CAD-like environment, instead of specifying geometry and thematic data using a text format. According to this, statements dealing with the survey data will be simple and will only reference survey data files, since the main work of executing the transaction would be conducted using a survey-data manipulation tool.

4.1.2. Back-End Tier

The execution of DSL statements will result in legal and survey dataset changes. The proposed system architecture solution contains a back-end tier that exposes services for LAS transaction execution. These services will present endpoints that could be accessed by different clients, such as standalone desktop applications, mobile device applications, web applications, or any other application that implements an appropriate client interface and logic.

Back-end services should provide several types of service:

- data retrieval;
- data validation;
- transaction execution.

In an example of ownership transfer, to insert a seller, the user should specify an identifier. To verify a person, the system should, for the identifier, retrieve seller data such as name, address, etc. The data validation service should check if that person has ownership over the spatial unit that is the subject of the transfer. Finally, the transaction execution service should initiate a data storage update.

On the other hand, the back-end tier should interact with the data-storage tier. Depending on the data-storage implementation, the back-end tier would provide appropriate functionality to initiate the execution of CRUD operations. If the data-storage tier has its own API for CRUD operations execution, the back-end tier should contain service consumer implementation. If data storage is in a relational database system, an object-relational mapping component should be provided. In any case, DSL for LAS statements storage functionality does not require any special features compared to other multi-tier systems.

4.1.3. Data Storage

It is expected that data should be stored in key-value pairs, where the key represents the identifier (id) of a transaction, while the value represents the LAS DSL statement. In this section, different possibilities for storing those key-value pairs, such as centralized ledgers (relational or NoSQL databases) or distributed ledgers (blockchain), will be discussed.

Most commonly, in an enterprise environment, data is stored in relational databases. Relational databases have been around for over 50 years with a lot of continuous development behind them. Relational databases have the support of another DSL, Structured Query Language (SQL), which is a powerful language to handle data and support strong consistency [57]. Relational databases work best with structured data that can "naturally" fit into tables [58].

Just over a decade ago, NoSQL databases appeared, and, unlike relational databases, one of the declared advantages is the possibility of the improved handling of unstructured data. Depending on the need, in NoSQL databases, data could be stored in key-values, documents and wide-columns [59], and graph stores. Another difference between relational and NoSQL databases is related to scalability. Relational databases are not suitable for horizontal scalability, while NoSQL databases can easily be horizontally scaled. In case more resources are needed in relational databases, that means that vertical scalability is needed. According to [60], even though NoSQL databases have been optimized for storing key-value data, not all NoSQL database implementations show better performance in instantiating, creating, reading, and deleting data, compared to relational databases.

Regarding scalability, storing DSL statements should not represent a significant amount of data within an LAS, so it is not expected that storing this data would be the main cause of the need to scale a database. With this in mind, between relational and NoSQL databases, relational databases should be the first pick for storing DSL statements, even though application of NoSQL in LAS is proposed in [61,62].

At about the same time as NoSQL, another technology appeared that was created for managing transactions, and that is blockchain technology. Blockchain technology is, together with big data and artificial intelligence, identified as a disruptive information-communication technology that will be the foundation of government 3.0 [63]. That is probably a reason why 10 out of 12 leading countries in e-government are specifically referencing the application of blockchain technology in this field [64]. It is clear that if blockchain and relational databases are compared regarding their performance related to instantiating, creating, reading, and deleting data, relational databases will win. Furthermore, blockchain only offers users the ability to create and read data, rather than the full set of CRUD operations. However, what it does offer could be of significance for storing LAS DSL statements. Blockchain represents the first implementation of distributed ledger technology, and, as such, it works in contrast to centralized (relational and NoSQL) ledgers that have centralized authorities. Blockchain data are stored on all full nodes that participate in the peer-to-peer network, while in centralized ledgers preserving data through backup is necessary. Data stored in a blockchain is transparent, and this can be a benefit in cases such as LAS where there is a significant number of stakeholders, while in centralized ledgers, the administrator can select which data can be publicly displayed. The existence of an administrator is probably the most significant difference between distributed and centralized ledgers. In centralized ledgers, it is possible to have a malicious actor that can change the data stored in the database. In undeveloped and developing countries, data misuse or data tampering as a result of corruption and fraud is identified as a big problem [65,66]. In a distributed ledger, malicious attacks are much harder to perform and usually require much more resources.

Since LAS DSL statements represent the starting point of LAS transactions, it would make sense to store them in a way that they cannot be altered by malicious actors anytime in the future, and this could be achieved by storing those key-value pairs on the blockchain. This could be achieved by implementing smart contracts on a blockchain. Smart contracts are computer programs that are deployed on the blockchain and are governed by the same set of rules that apply to all blockchain transactions. Once stored, those records could be used as a "source of truth", and, in case any transactions in a centralized ledger are under suspicion, they could be compared with DSL data stored in a blockchain that should have been its starting point.

Since the blockchain is not suitable for storing high volumes of data, full LAS implementation could be achieved through a hybrid form, as suggested in [67,68], where traditional systems, such as centralized ledgers, are combined with smart contracts. The application of smart contracts for managing LAS transactions is also discussed in [69], where an example of a smart contract for governing those transactions is presented.

5. Proposed Solution

The proper way to design and implement a DSL is to create abstract syntax, concrete syntax, and semantics. Abstract syntax represents a basic skeleton of a language and can be used as a starting point for defining concrete syntax and semantics, but a concrete order of steps could be argued [70]. In this case, a first-language grammar was created, and, based on that grammar, a meta-model was built. Meta-modeling represents a popular method for defining the abstract syntax of a language. The grammar of DSL for LAS transactions was created using textX. TextX (The full documentation on textX could be found on the following link (http://textx.github.io/textX/3.0/), accessed 28 June 2022) is a meta-language and a tool that was created to facilitate the building of DSLs. It was built on top of the Arpeggio parser, and it makes it possible to construct both the Arpeggio parser and meta-model from a single grammar description at run-time [71].

The TextX grammar description represents a set of rules. Rules could be one of common rules (or simply rules), abstract rules, or match rules. The common rules contain at least one assignment, the abstract rules have no assignments reference at least one abstract or common rule, and the match rules are the basic building blocks for more complex expressions—they will consume input on success. Each rule defines one concept from a meta-model and at the same time a syntax for that concept. A rule begins with a name and column symbol and ends with a semi-column.

In Listing 1, the abstract rules for File, Transaction, LegalTransaction, and GeoTransaction are declared. An example of a rule declaration can be seen in lines 5 through 7. In line 5, the abstract rule name is declared with the word File followed by a column. In line 6, it is stated that in File there could be zero or more transactions of type Transaction, indicated by the asterisk symbol. In line 7, there is a semi-column symbolizing the end of the specific rule. In a similar way, the abstract rule Transaction is declared, stating that the transaction is either a LegalTransaction or GeoTransaction. Abstract rule LegalTransaction declares that a legal transaction is one of Transfer, Update, or Create, while abstract rule GeoTransaction states that survey transaction is either CreateDevSite or ApplyDevSite.

Listing 1. DSL textX grammar—File, Transaction, LegalTransaction, and GeoTransaction.

```
1  /*
2      Land Administration DSL textX grammar
3  */
4
5  File:
6      transactions*=Transaction
7  ;
8
9  Transaction:
10     LegalTransaction | GeoTransaction
11 ;
12
13 LegalTransaction:
14     Transfer | Update | Create
15 ;
16
17 GeoTransaction:
18     CreateDevSite | ApplyDevSite
19 ;
```

In Listing 2, the Transfer, Party, Parcel, Building, and Share rules are presented. In the Transfer rule, it is declared that transfers should have one or more parties selling one or more parcels with specific shares to one or more parties in a specific share. A Share is an optional rule, since, in case no share is specified, it is considered that full ownership is being transferred. The Party rule is defined with regular expressions to make it possible for a party to be represented with strings and/or numbers. Similarly, the declarations of Parcel, Building, and Share define common formats for representing parcel, building, and share in LASs. For example, in the case of the parcel 34/101/2, it would mean that, in the cadastral municipality with id 34, there is a parcel with id 101, with a parcel part with id 2. In the case of the building, 37/104/3/10, 37 represents the cadastral municipality with the same id, 104 represents the id of a parcel in that cadastral municipality, 3 represents the building id, and 10 the unit id. For a share, a fraction is used to represent a share of the ownership where 1/2 would mean that the party is participating in the transfer with a 50% share of the ownership.

Listing 2. DSL textX grammar—Transfer, Party, Parcel, Building, and Share.

```
21 Transfer:
22     'transfer'
23     'from'
```

```
        _from+=Party[',']
        'parcel' (parcel_all?='all' | parcel=Parcel)
        ('share' from_share+=Share[','])?
    'to'
        to+=Party[',']
        ('share' to_share+=Share[','])?
;

Party: /[0-9a-zA-Z_-]+/;
Parcel: ko=INT '/' parcel=INT ('/' part=INT)?;
Building: ko=INT '/' parcel=INT '/' building=INT ('/' unit=INT)?;
Share: nom=INT '/' den=INT;
```

Listing 3, presents abstract rules for Update and Right, together with rules for UpdateParty, UpdateUnit, and Mortgage. The Update abstract rule states that the update falls under either the UpdateParty rule or the UpdateUnit rule. Rule UpdateParty declares that the properties (i.e., first name, last name, name, and address) of a party could be updated, while rule UpdateUnit declares that the property's land use and right could be updated for parcel or building. The abstract rule Right states that it is related to mortgage, and the Mortgage rule defines the properties of amount and interest rate of a mortgage.

Listing 3. DSL textX grammar—Update, UpdateParty and UpdateUnit.

```
Update:
    UpdateParty | UpdateUnit
;

UpdateParty:
    'update'
    'party' party=Party
    (('firstName' first_name=STRING)?
     ('lastName' last_name=STRING)?
     ('name' name=STRING)?
     ('address' address=STRING)?
    )
;

UpdateUnit:
    'update'
    ('parcel' parcel=Parcel
     | 'building' building=Building)
    ('landUse' land_use=ID
     |'right' right=Right 'party' party=Party)
;

Right:
    Mortgage;

Mortgage: 'mortgage' ammount=FLOAT percent=FLOAT '%';
```

In Listing 4, the Create rule is specified for adding/creating a new building. For adding a new building, a building id is necessary, as well as the specification of one or more parties that posses a specific share of the ownership.

Listing 4. DSL textX grammar—Create.

```
Create:
    'create'
     'building' building=Building
     'party' party+=Party[',']
     'share' share+=Share[',']
;
```

In Listing 5, the rules for CreateDevSite, ApplyDevSite, DevSiteID, GeoAggregateID, and Comment are declared. CreateDevSite is a rule for performing survey transactions. It states that one or more geoaggregate should be selected, and a development site created.

Since this is related to the manipulation of the spatial objects, it is not feasible to create a textual DSL for these actions. Instead, a GUI application should be launched, and selected geoaggregates loaded. Upon making the changes in the spatial data using this GUI, the ApplyDevSite rule is used to initiate the process of storing the changes in the legal register. The remaining rules, DevSiteID, GeoaggregateID, and Comment, declare how the development site and geoaggregate ids should be formatted, as well as how comments could be added to the proposed DSL.

Listing 5. DSL textX grammar—CreateDevSite, ApplyDevSite, DevSiteID, and GeoAggregateID.

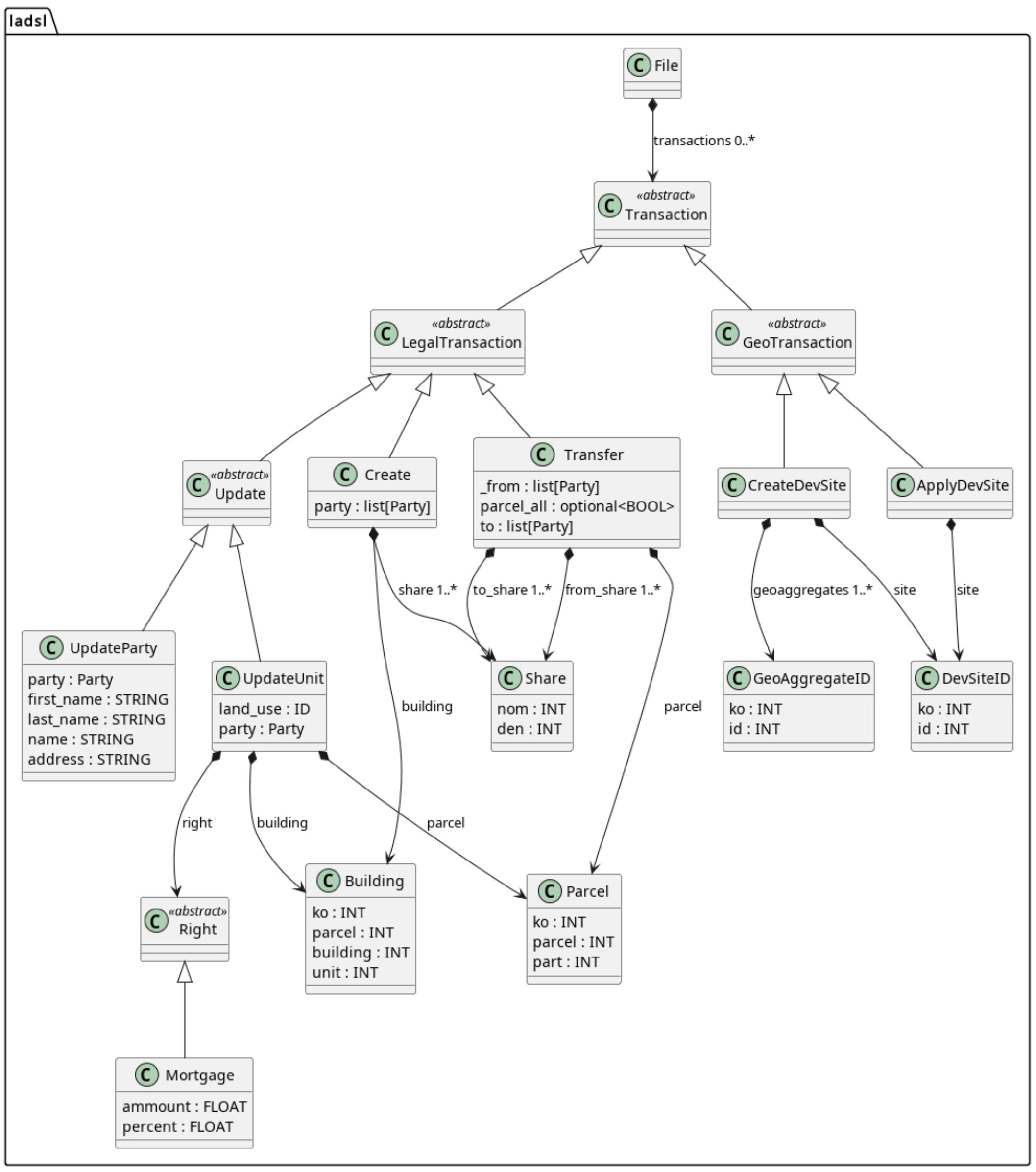
```
CreateDevSite:
    'create' 'development' 'site' site=DevSiteID
    'geoaggregates' geoaggregates+=GeoAggregateID[',']
;

ApplyDevSite:
    'apply' 'development' 'site' site=DevSiteID
;

DevSiteID:
    ko=INT '/' id=INT;

GeoAggregateID:
    ko=INT '/' id=INT;

// special rule for comments.
Comment:
    /\/\/.*$/
;
```

Based on the described textX grammar, a meta-model for DSL is created; it is presented in Figure 7.

To demonstrate the usage of the proposed DSL, several use cases were created. For each use case, a test was written. In Listing 6, a DSL for simple transfer where Party1 is selling parcel 23/101 in share 1/1 to Party2 is shown in lines 11 through 16.

Listing 6. DSL test—Simple transfer.

```
1  import pytest
2  from textx import~metamodel_for_language
3
4  @pytest.fixture
5  def mm():
6      return metamodel_for_language('ladsl')
7
8  def test_transfer(mm):
9
10     model = '''
11     transfer
12        from party1
13        parcel 23/101
14        share 1/1
15        to party2
16        share 1/1
17     '''
18
19     m = mm.model_from_str(model)
20
21     assert len(m.transactions) == 1
22     t = m.transactions[0]
23     assert t._from[0] == 'party1'
24     assert t.to[0] == 'party2'
25     assert (t.from_share[0].nom, t.from_share[0].den) == (1, 1)
26     assert (t.to_share[0].nom, t.to_share[0].den) == (1, 1)
```



Match rules:

Name	Rule details
Party	[0-9a-zA-Z_-]+
Comment	VV.*$

Figure 7. Meta-model of DSL for LA.

In Listing 7, party11 and party12 are selling parcel 23/101 with the share in the ownership of 1/2 and 1/2, to party21 and party22 in the shares 1/3 and 2/3, respectively. The DSL for this transfer is written in lines 31 through 36.

Listing 7. DSL test—Transfer involving multiple parties and different share of ownership.

```
28 def test_transfer_multi(mm):
29
30     model = '''
31     transfer
32        from party11, party12
33        parcel 23/101
34        share 1/2, 1/2
35        to party21, party22
36        share 1/3, 2/3
37     '''
```

```
38
39      m = mm.model_from_str(model)
40
41      assert len(m.transactions) == 1
42      t = m.transactions[0]
43      assert len(t._from) == len(t.to) == 2
44      assert t._from[0] == 'party11'
45      assert t._from[1] == 'party12'
46      assert t.to[0] == 'party21'
47      assert t.to[1] == 'party22'
48      assert (t.from_share[0].nom, t.from_share[0].den) == (1, 2)
49      assert (t.from_share[1].nom, t.from_share[1].den) == (1, 2)
50      assert (t.to_share[0].nom, t.to_share[0].den) == (1, 3)
51      assert (t.to_share[1].nom, t.to_share[0].den) == (2, 3)
```

In Listing 8, the first and last names of the party1 are changed. The DSL for this update is written in lines 50 through 53.

Listing 8. DSL test—Updating party.

```
53  def test_update_party_1(mm):
54      model = '''
55      update
56          party party1
57          firstName "John"
58          lastName "Smith"
59      '''
60
61      m = mm.model_from_str(model)
62
63      assert len(m.transactions) == 1
64      t = m.transactions[0]
65      assert t.party == 'party1'
66      assert t.first_name == 'John'
67      assert t.last_name == 'Smith'
```

In Listing 9, the company name of party1 is changed. The DSL for this update is written in lines 66 through 68.

Listing 9. DSL test—Updating party representing a company.

```
69  def test_update_party_2(mm):
70      model = '''
71      update
72          party party1
73          name "New Company Name"
74      '''
75
76      m = mm.model_from_str(model)
77
78      assert len(m.transactions) == 1
79      t = m.transactions[0]
80      assert t.party == 'party1'
81      assert t.name == 'New Company Name'
```

In Listing 10, party1's address is changed. The DSL for this update is written in lines 80 through 82.

Listing 10. DSL test—Updating party address.

```
83  def test_update_party_3(mm):
84      model = '''
85      update
86          party party1
87          address "New Address"
88      '''
89
90      m = mm.model_from_str(model)
```

```
91
92      assert len(m.transactions) == 1
93      t = m.transactions[0]
94      assert t.party == 'party1'
95      assert t.address == 'New Address'
```

In Listing 11, land use for part of parcel 34/101/2 is changed. The DSL for this update is written in lines 94 through 96.

Listing 11. DSL test—Updating parcel land use.

```
97  def test_update_parcelpart_1(mm):
98      model = '''
99      update
100         parcel 34/101/2
101         landUse vineyard
102     '''
103
104     m = mm.model_from_str(model)
105
106     assert len(m.transactions) == 1
107     t = m.transactions[0]
108     assert t.parcel.ko == 34
109     assert t.parcel.parcel == 101
110     assert t.parcel.part == 2
111     assert t.land_use == 'meadow'
```

In Listing 12, the mortgage is set at 45,000.00 with a 7% interest rate, on building 34/101/2 on behalf of party1. The DSL for this update is written in lines 110 through 114.

Listing 12. DSL test—Updating rights.

```
113 def test_update_right(mm):
114     model = '''
115     update
116         building 34/101/2
117         right
118           mortgage 45,000.00 7%
119         party party1
120     '''
121
122     m = mm.model_from_str(model)
123
124     assert len(m.transactions) == 1
125     t = m.transactions[0]
126     assert t.building.ko == 34
127     assert t.building.parcel == 101
128     assert t.building.building == 2
129     assert t.right.ammount == 45,000
130     assert t.right.percent == 7
```

In Listing 13, a new building is added/created, with id 34/101/2, and party1 and party2 are set as the owners in 1/3 and 2/3 shares. The DSL for this creation is written in lines 129 through 132.

Listing 13. DSL test—Creating building.

```
132 def test_create_building(mm):
133     model = '''
134     create
135         building 34/101/2
136         party party1, party2
137         share 1/3, 2/3
138     '''
139
140     m = mm.model_from_str(model)
141
142     assert len(m.transactions) == 1
```

```
143     t = m.transactions[0]
144     assert t.building.ko == 34
145     assert t.building.parcel == 101
146     assert t.building.building == 2
147     assert t.party == ['party1', 'party2']
148     assert t.share[0].nom == 1
149     assert t.share[0].den == 3
150     assert t.share[1].nom == 2
151     assert t.share[1].den == 3
```

In Listing 14, a new development with id 34/77 is being created from geoaggregates 34/17 and 34/18. The DSL for this creation is written in lines 149 through 150.

Listing 14. DSL test—Creating development site.

```
153 def test_create_site(mm):
154 
155     model = '''
156     create development site 34/77
157     geoaggregates 34/17, 34/18
158     '''
159 
160     m = mm.model_from_str(model)
161 
162     assert len(m.transactions) == 1
163     t = m.transactions[0]
164     assert t.site.ko == 34 and t.site.id == 77
165 
166     assert len(t.geoaggregates) == 2
167     g = t.geoaggregates[0]
168     assert g.ko == 34 and g.id == 17
169 
170     assert len(t.geoaggregates) == 2
171     g = t.geoaggregates[1]
172     assert g.ko == 34 and g.id == 18
```

In Listing 15, the finished development site with id 34/77 is being applied to the legal register. The DSL for this apply statement is written in line 170.

Listing 15. DSL test—Applying development site.

```
174 def test_apply_site(mm):
175 
176     model = '''
177     apply development site 34/77
178     '''
179 
180     m = mm.model_from_str(model)
181 
182     assert len(m.transactions) == 1
183     t = m.transactions[0]
184     assert t.site.ko == 34 and t.site.id == 77
```

The full code can be found at the following link (https://github.com/djordjeprzulj/ladsl, accessed 28 June 2022).

6. Conclusions

In this paper, LAS transactions have been analyzed and discussed. Common challenges regarding LAS transactions' execution and LAS data integrity enforcement have been addressed. A use case in which a survey data update initialises a change in legal data is also presented. Automation of this part of the process would improve LAS transaction execution.

As previously mentioned, to the best knowledge of the authors, there was no research conducted on developing DSL in the field of LASs or, more specifically, for managing LAS transactions. Several papers related to DSLs in domains of modeling processes in landscapes and land use do not have either defined meta-models, grammar, or test cases.

A DSL for LAS transactions would introduce most of the benefits that the application of DSL brings, such as a new level of abstraction, domain expert usage, statement validation, statement interpretation on various platforms, and using different GPLs. A proposal for a system architecture that would enable the writing, parsing, and execution of DSL for LAS transactions statements is also presented in the paper. The grammar of the language is presented together with meta-model visualisation. Examples of statements and tests are also given for representative types of statements.

However, the proposed DSL for LAS transactions has some limitations. The biggest one is that textual DSL is not suitable for geometry data manipulation. To effectively prepare data for survey transactions, an appropriate GUI tool should be provided. The development of domain-specific tool for survey data manipulation is a significant area for further research and development. This tool should guide the end user through the process of survey data preparation for transaction execution, so the resulting data are valid, correct, and accurate.

Another further research direction is the development of a user-friendly integrated development environment (IDE) that would assist domain experts in writing DSL for LAS transactions statements. Typical IDE tools, such as syntax highlighting, code completion, or code templates, would make users more efficient and code less error prone.

Aspects of DSL for LAS transactions that should be further analyzed are possible alternative options for statement storage. In the case of the distributed ledger, private or hybrid blockchain should be considered together with the security aspects and context of the corresponding legal framework.

Author Contributions: Conceptualization, Đorđe Pržulj and Igor Dejanović; Methodology, Igor Dejanović; Software, Igor Dejanović; Validation, Miroslav Stefanović and Teodora Lolić; Writing—Original Draft Preparation, Đorđe Pržulj, Miroslav Stefanović, Teodora Lolić and Srđan Sladojević; Writing—Review & Editing, Đorđe Pržulj, Miroslav Stefanović and Srđan Sladojević; Visualization, Đorđe Pržulj, Igor Dejanović, and Miroslav Stefanović; Supervision, Đorđe Pržulj; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Henssen, J.G.; Williamson, I.P. Land Registration, Cadastre and Its Interaction–A World Perspective. In Proceedings of the FIG XIX Congress, Helsinki, Finland, 10 June 1990; pp. 14–43.
2. Petronijevic, M.; Višnjevac, N.; Pračević, N.; Bajat, B. The Extension of IFC For Supporting 3D Cadastre LADM Geometry. *ISPRS Int. J. Geoinf.* **2021**, *10*, 297. [CrossRef]
3. Paasch, J.; van Oosterom, P.; Lemmen, C.; Paulssond, J. Further Modelling of LADM's Rights, Restrictions and Responsibilities (RRRs). *Land Use Policy* **2015**, *49*, 680–689 [CrossRef]
4. van Oosterom, P.; Lemmen, C. The Land Administration Domain Model (LADM): Motivation, Standardisation, Application and Further Development. *Land Use Policy* **2015**, *49*, 527–534 [CrossRef]
5. Bennett, T.; Rajabifard, A.; Williamson, I.; Wallace, J. On the Need for National Land Administration Infrastructures. *Land Use Policy* **2012**, *29*, 208–219. [CrossRef]
6. Çağdaş, V.; Stubkjær, E. Core Immovable Property Vocabulary for European Linked Land Administrations. *Surv. Rev.* **2015**, *47*, 49–60. [CrossRef]
7. Iban, M.; Aksu, O. A Model for Big Spatial Rural Data Infrastructure in Turkey: Sensor-driven and Integrative Approach. *Land Use Policy* **2020**, *91*, 104376. [CrossRef]
8. Inan, H. Associating Land Use/Cover Information with Land Parcels Represented in LADM. *Land Use Policy* **2015**, *49*, 626–633. [CrossRef]

9. Drobež, P.; Kosmatin Fras, M.; Ferlan, M.; Lisec, A. Transition From 2D to 3D Real Property Cadastre: The Case of the Slovenian Cadastre. *Comput. Environ. Urban. Syst.* **2015**, *62*, 125–135. [CrossRef]
10. Przewięźlikowska, A. Legal Aspects of Synchronising Data on Real Property Location in Polish Cadastre and Land and Mortgage Register. *Land Use Policy* **2020**, *95*, 104606. [CrossRef]
11. Cetl, V.; Roic, M.; Mastelic Ivic, M. Towards a real property Cadastre in Croatia. *Surv. Rev.* **2012**, *44*, 17–22. [CrossRef]
12. Kitsakis, D.; Apostolou, C.; Dimopoulou, E. Three-dimensional cadaster modelling of customary real property rights. *Surv. Rev.* **2015**, *50*, 107–121. [CrossRef]
13. Kara, A.; Çağdaş, V.; Isikdag, U.; van Oosterom, P.; Lemmend, C.; Stubkjaer, E. The LADM Valuation Information Model and Its Application to the Turkey Case. *Land Use Policy* **2021**, *104*, 105307. [CrossRef]
14. Tomić, H.; Mastelić Ivić, S.; Roić, M.; Šiško, J. Developing an Efficient Property Valuation System Using the LADM Valuation Information Model: A Croatian Case Study. *Land Use Policy* **2021**, *104*, 105368. [CrossRef]
15. Unger, E.M.; Zevenbergen, J.; Bennett, R.; Lemmen, C. Application of LADM For Disaster Prone Areas and Communities. *Land Use Policy* **2019**, *180*, 118–126 [CrossRef]
16. Unger, E.M.; Bennett, R.; Lemmen, C.; Zevenbergen, J. LADM for Sustainable Development: An Exploratory Study on the Application of Domain-specific Data Models to Support the SDGs. *Land Use Policy* **2021**, *108*, 105499. [CrossRef]
17. Rockson, G.; Bennett, R.; Groenendijk, L. Land Administration for Food Security: A Research Synthesis. *Land Use Policy* **2013**, *32*, 337–342 [CrossRef]
18. Zysk, E.; Dawidowicz, A.; Nowak, M.; Figurska, M.; Źróbek, S.; Źróbek, R.; Burandt, J. Organizational Aspects of the Concept of a Green Cadastre for Rural Areas. *Land Use Policy* **2020**, *91*, 104373. [CrossRef]
19. Habib, M. Developing a Sustainability Strategy for Multipurpose Cadastre in Post-Conflict Syria. *Land Use Policy* **2020**, *97*, 104782. [CrossRef]
20. Indrajit, A.; van Loenen, B.; Ploeger, H.; van Oosterom, P. Developing a Spatial Planning Information package in ISO 19152 Land Administration Domain Model. *Land Use Policy* **2020**, *98*, 104111. [CrossRef]
21. Kaufmann, J.; Steudler, D. *Cadastre 2014—A Vision for a Future Cadastral System*; International Federation of Surveyors (FIG): Copenhagen, Denmark, 1998.
22. *ISO 19152:2012*; Geographic Information—Land Administration Domain Model (LADM). Technical Committee ISO/TC 211; International Organization for Standardization (ISO): Geneva, Switzerland, 2012.
23. Bennett, R.; Rajabifard, A.; Kalantari, M.; Wallace, J.; Williamson, I. Cadastral Futures: Building a New Vision for the Nature and Role of Cadastres. In Proceedings of the Fig Congress, Sydney, Australia, 11 April 2010.
24. Lemmens, M. Towards Cadastre 2034. Available online: https://www.gim-international.com/content/article/towards-cadaster-2034 (accessed on 22 July 2022).
25. Lemmen, C.H.J.; Unger, E.; van Oosterom, P.J.M.; Kalantari, M.; De Zeeuw, K. Exploring Options for Standardisation of Processes and Transactions in Land Administration. In Proceedings of the World Bank Land and Poverty Conference 2018: Land Governance in an Interconnected World, Washington, DC, USA, 19 March 2018.
26. Pržulj, Đ.; Radaković, N.; Sladić D.; Radulović, A.; Govedarica, M. Domain Model for Cadastral Systems with Land Use Component. *Surv. Rev.* **2017**, *51*, 135–146. [CrossRef]
27. Stefanović, M.; Pržulj, Đ.; Stefanović, D.; Vukmanović, M.; Ristić, S. OCL Specification of Inter-Register Integrity Constraints in Land Administration Systems. In Proceedings of the Central European Conference on Information and Intelligent Systems, Varaždin, Croatia, 27 September 2017; pp. 273–281.
28. Vučić, N.; Markovinović, D.; Mičević, B. LADM in the Republic of Croatia–Making and Testing Country Profile. In Proceedings of the 5th Land Administration Domain Model Workshop, Kuala Lumpur, Malaysia, 24 September 2013; pp. 329–344.
29. *ISO 19157:2013*; Geographic Information—Data Quality. Technical Committee ISO/TC 211; International Organization for Standardization (ISO): Geneva, Switzerland, 2013
30. *ISO 19115-1:2014*; Geographic Information—Metadata—Part 1: Fundamentals. Technical Committee ISO/TC 211; International Organization for Standardization (ISO): Geneva, Switzerland, 2014
31. *ISO 19115-2:2019*; Geographic Information—Metadata—Part 2: Extensions for Acquisition and Processing. Technical Committee ISO/TC 211; International Organization for Standardization (ISO): Geneva, Switzerland, 2019.
32. Voelter, M. Introduction to DSLs. Available online: http://dslbook.org/ (accessed on 20 March 2022).
33. Tomassetti, F. Domain Specific Languages for Smart Contracts. Available online: https://tomassetti.me/domain-specific-languages-for-contacts/ (accessed on 29 May 2022).
34. *ISO 19106:2004*; Geographic information—Profiles. Technical Committee ISO/TC 211; International Organization for Standardization (ISO): Geneva, Switzerland, 2004
35. Radulović, A.; Sladić, D.; Govedarica, M.; Ristić, A.; Jovanović, D. LADM Based Utility Network Cadastre in Serbia. *ISPRS Int. J. Geo-Inf.* **2019**, *8*, 206. [CrossRef]
36. Lisjak, J.; Roić, M.; Tomić H.; Mastelić Ivić, S. Croatian LADM Profile Extension for State-Owned Agricultural Land Management. *Land* **2019**, *10*, 222. [CrossRef]
37. Abdeslam Adad, M.; El Hassane, S.; El-ayachi, M.; Ibannaina, F. Supporting Land Data Integration and Standardization Through the LADM Standard: Case of Morocco's Country Profile MA-LADM. *Land Use Policy* **2020**, *97*, 104762. [CrossRef]

38. Lee, B.-M.; Kim, T.-J.; Kwak, B.-Y.; Lee, Y.-H.; Choi, J. Improvement of the Korean LADM Country Profile to Build a 3d Cadastre Model. *Land Use Policy* **2015**, *49*, 660–667. [CrossRef]
39. Kalogianni, E.; Kalantari, M.; Dimopoulou, E.; van Oosterom, P.J.M. LADM Country Profiles Development: Aspects to Be Reflected and Considered. In Proceedings of the 8th Land Administration Domain Model Workshop (LADM 2019), Kuala Lumpur, Malaysia, 1 October 2019.
40. Lemmen, C.H.J.; van Oosterom, P.J.M.; Kalantari, M. Towards a New Working Item Proposal for Edition II of LADM. In Proceedings of the 7th International FIG Workshop on the Land Administration Domain Model, Zagreb, Croatia, 12 April 2018.
41. Polat, A.A.; Alkan, M. Design and Implementation of a LADM-based External Archive Data Model for Land Registry and Cadastre Transactions in Turkey: A Case Study of Municipality. *Land Use Policy* **2018**, *77*, 249–266. [CrossRef]
42. Lisec, A.; Ferlan, M.; Šumrada, M. UML Notation for the Rural Land Transaction Procedure. *Geod. Vestn.* **2007**, *51*, 11–34.
43. Kemoe, M. *The Land Registry in the Blockchain–Testbed*; Kairos Future: Stockholm, Sweden, 2017
44. *ISO 19103:2005*; Geographic Information—Conceptual Schema Language. Technical Committee ISO/TC 211; International Organization for Standardization (ISO): Geneva, Switzerland, 2005.
45. Mernik, M.; Heering, J.; Sloane, A.M. When and How to Develop Domain-Specific Languages. *ACM Comput. Surv.* **2005**, *37*, 316–344. [CrossRef]
46. Poltronieri, I.; Zorzo, A.F.; Bernardino, M.; Medeiros, B.; de Borba Campos, M. Heuristic Evaluation Checklist for Domain-specific Languages. In Proceedings of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021), Online, 6 February 2021; pp. 37–48.
47. van Deursen, A.; Klint, P.; Visser, J. Domain-Specific Languages: An Annotated Bibliography. *ACM Sigplan Not.* **2000**, *35*, 26–36. [CrossRef]
48. Degenne, P.; Lo Seen, D.; Parigot, D.; Forax, R.; Tran, A.; Ait Lahcen, A.; Curé, O.; Jeansoulin, R. Design of a Domain Specific Language for Modelling Processes in Landscapes. *Ecol. Model.* **2009**, *220*, 3527–3535. [CrossRef]
49. Fall, A.; Fall, J. A Domain-Specific Language for Models of Landscape Dynamics. *Ecol. Model.* **2001**, *141*, 1–18. [CrossRef]
50. Gaucherel, C.; Giboire, N.; Houet, T.; Baudry, J.; Burel, F. A Domain-specific Language for Patchy Landscape Modelling: The Brittany Agricultural Mosaic as a Case Study. *Ecol. Model.* **2006**, *194*, 233–243. [CrossRef]
51. Grueau, C.; Araújo, J. Towards a Domain Specific Modeling Language for Agent-based Models in Land Use Science. In Proceedings of the 28th Annual ACM Symposium on Applied Computing, Coimbra, Portugal, 18 March 2013; pp. 83–85.
52. de Sousa, L.M.; da Silva, A.R. A Domain Specific Language for Spatial Simulation Scenarios. *Geoinformatica* **2016**, *20*, 117–149. [CrossRef]
53. Zevenbergen, J.; De Vries, W.; Bennett, R. *Advances in Responsible Land Administration*; CRC Press: Boca Raton, FL, USA, 2016.
54. Williamson, I.; Enemark, S.; Wallace, J.; Rajabifard, A. *Land Administration for Sustainable Development*; ESRI Press: Boca Raton, FL, USA, 2010.
55. Gamma, R.; Helm, R.; Vlissides, J.; Johnson, R. *Design Patterns: Elements of Reusable Object-Oriented Software*; Addison-Wesley: Boston, MA, USA, 1994
56. Hjelmblom, M.; Paasch, J.; Paulsson, J.; Edlund, M.; Bökman, M. Towards Automation of the Swedish Property Formation Process: A Structural and Logical Analysis of Property Subdivision. *NJSR* **2019**, *14*, 19–63. [CrossRef]
57. Wang, Z.; Wei, Z.; Liu, H. Research on High Availability Architecture of SQL and NoSQL. In Proceedings of the AIP Conference, Wuhan, China, 25 February 2017
58. Leavitt, N. Will NoSQL Databases Live Up to Their Promise? *Computer* **2010**, *43*, 12–14. [CrossRef]
59. Cattell, R. Scalable SQL and NoSQL Data Stores. *SIGMOD Rec.* **2010**, *37*, 12–27. [CrossRef]
60. Li, Y.; Manoharan, S. A Performance Comparison of SQL and NoSQL Databases. In Proceedings of the IEEE 2013 Pacific Rim Conference on Communications, Computers and Signal Processing, Victoria, BC, Canada, 27 August 2013
61. Višnjevac, N.; Šoškić, M.; Mohajlović, R.; Cvjetinović, Ž. Using NoSQL databases in the 3D cadaster domain. *Geod. Vestn.* **2017**, *61*, 412–426. [CrossRef]
62. Višnjevac, N.; Mihajlović, R.; Šoškić, M.; Cvjetinović, Ž.; Bajat, B. Prototype of the 3D Cadastral System Based on a NoSQL Database and a JavaScript Visualization Application. *SPRS Int. J. Geoinf.* **2019**, *8*, 227. [CrossRef]
63. Pereira, G.V.; Charalabidis, Y.; Alexopoulos, C.; Mureddu, F.; Parycek, P.; Ronzhyn, A.; Sarantis, D.; Flak, L.; Wimmer, M.A. Scientific Foundations Training and Entrepreneurship Activities in the Domain of ICT-enabled Governance. In Proceedings of the 19th Annual International Conference on Digital Government Research: Governance in the Data Age, Delft, The Netherlands, 30 May 2018.
64. United Nations Economic Commission for Europe (UNECE) Working Party on Land Administration (WPLA). *Survey on Land Administration Systems*; United Nations: Geneva, Switzerland, 2014
65. Vos, J. Blockchain-based Land Registry: Panacea, Illusion or Something in Between? In Proceedings of the IPRA/CINDER Congress, Dubai, United Arab Emirates, 22 February 2016.
66. Lemieux, V. Trusting Records: Is Blockchain Technology the Answer? *Rec. Manag. J.* **2016**, *26*, 110–139. [CrossRef]
67. Bennett, R.; Miller, T.; Pickering, M.; Kara, A. Hybrid Approaches for Smart Contracts in Land Administration: Lessons from Three Blockchain Proofs-of-Concept. *Land* **2021**, *10*, 220. [CrossRef]
68. Sladić, G.; Milosavljević, B.; Nikolić, S.; Sladić, D.; Radulović, A. A Blockchain Solution for Securing Real Property Transactions: A Case Study for Serbia. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 35. [CrossRef]

69. Stefanović, M.; Pržulj, Ð.; Ristić, S.; Stefanović, D.; Nikolić, D. Smart Contract Application for Managing Land Administration System Transactions. *IEEE Access* **2022**, *10*, 39154–39176. [CrossRef]
70. Andova, S.; van den Brand, M.G.J.; Engelen, L.J.P.; Verhoeff, T. MDE Basics with a DSL Focus. In *Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2012
71. Dejanović, I.; Vaderna, R.; Milosavljević, G.; Vuković, Ž. TextX: A Python tool for Domain-Specific Languages Implementation. *Knowl.-Based Syst.* **2017**, *115*, 1–4. [CrossRef]