UDC 81’ 33

DOi 10.30837/ bi.2025.1(102).01

Alona Dorozhynska[1], Sergiy Dorozhynskyy[2]

[1]Ukrainian Lingua-Information Fund of NAS of Ukraine, Kyiv, Ukraine,
alonadrzh@gmail.com, ORCID iD: 0000-0001-6554-6731

[2]Kyiv University of Intellectual Property and Law, Kyiv, Ukraine,
dorozhun1706@gmail.com, ORCID iD: 0000-0002-5395-6423

FROM DICTIONARY TO DATABASE: TRANSFORMATION AND PROTECTION

The article outlines the importance of data digitization in modern society and in the context of countering the aggressor. It also shows the problems associated with paper-based media that lose their relevance and become unsuitable for work over time. The study proposes to use LiteDB as a lightweight NoSQL database for digitizing information due to its ease of use, embeddability, JSON/BSON data format, security assessment, and easy scalability. Particular emphasis is placed on the use of the HTMLAgilityPack library for working with HTML in the .NET environment. It is noted that it helps to parse HTML pages, extract structured data, and convert HTML to a structured format. An important priority of these tools is the ability to use built-in or connected information security systems, the best option of which is quantum cryptography and quantum key distribution. The advantage of such protection is its resistance to most modern attacks, even under conditions of uncertainty.

DATA DIGITIZATION, LITEDB, NOSQL DATABASE, HTMLAGILITYPACK, QUANTUM CRYPTOGRAPHY, QUANTUM KEY DISTRIBUTION PROTOCOLS, INFORMATION SECURITY, PAPER MEDIA, DATABASE (DB), DATABASE ARCHITECTURE, JSON/BSON FORMAT, PARSING, XML

А. Дорожмнська, С. Дорожинський. Від словника до бази даних: трансформація та захист. У статті окреслено важливість оцифрування даних у сучасному суспільстві та в контексті протидії агресору. Також показано проблеми, пов'язані з паперовими носіями, які з часом втрачають свою актуальність і стають непридатними для роботи. У дослідженні пропонується використовувати litedb як легку nosql-базу даних для оцифрування інформації завдяки простоті використання, можливості вбудовування, формату даних json/bson, оцінці безпеки та легкій масштабованості. Особливий акцент зроблено на використанні бібліотеки htmlagilitypack для роботи з html в середовищі .net. Зазначається, що вона допомагає аналізувати html-сторінки, витягувати структуровані дані та конвертувати html у структурований формат. Важливим пріоритетом цих інструментів є можливість використання вбудованих або підключених систем захисту інформації, найкращим варіантом яких є квантова криптографія та квантовий розподіл ключів. Перевагою такого захисту є його стійкість до більшості сучасних атак, навіть в умовах невизначеності.

ОЦИФРУВАННЯ ДАНИХ, LITEDB, БАЗА ДАНИХ NOSQL, HTMLAGILITYPACK, КВАНТОВА КРИПТОГРАФІЯ, ПРОТОКОЛИ КВАНТОВОГО РОЗПОДІЛУ КЛЮЧІВ, ІНФОРМАЦІЙНА БЕЗПЕКА, ПАПЕРОВІ НОСІЇ, БАЗА ДАНИХ (DB), АРХІТЕКТУРА БАЗИ ДАНИХ, ФОРМАТ JSON/BSON, ПАРСИНГ, XML

Introduction

An important solution for society in the current conditions of existence and counteraction to the aggressor is the digitization of data in paper format. The problem is that paper media lose their relevance, are little used, and become unsuitable for work and processing over time. Their digital counterparts are more time-resistant and more accessible to the user. One of the most important stages of the digitization process is the formation of a database, its architecture and content. It should be remembered that paper-based media are usually unstructured or semi-structured. Therefore, databases that are easy to use and fill are suitable for such tasks. That's why we chose LiteDB, a lightweight NoSQL database designed for use in .NET applications, including C# and .NET Core. It can be an attractive option for some digitized data tasks for the following reasons:

– Ease of use: LiteDB has a simple API and integrates easily with .NET projects. Its syntax and data structure are well suited to working with C#.

– Embeddability: LiteDB can be easily embedded into an application because it does not require a separate database server to be installed. You just need to add the library to the project and start using it.

– JSON/BSON data format: LiteDB uses the BSON (Binary JSON) data format, which makes it efficient for working with JSON objects. This is especially useful if you need to work with unstructured or semi-structured data.

– Security assessment: LiteDB has support for database encryption, which can be an important aspect for security, as it is possible to protect data with custom encryption keys, as well as apply the latest security methods, such as quantum cryptography and quantum key distribution.

– Easy scalability: In cases where the project requires a small database or does not require complex operations, LiteDB can be the best option due to its lightness and efficiency.

The use of this database makes it possible to use a large number of libraries to simplify the process of data digitization. One of them is HTMLAgilityPack, a library for working with HTML in the .NET environment. Its main purpose is to parse and process HTML documents. In the context of digitized data and databases, HTMLAgilityPack can be useful for extracting data before

saving it to a database. The main aspects of the benefits of using this library:

– Parsing HTML pages: HTMLAgilityPack provides a convenient way to access various elements of an HTML page, extracting text, attributes, and other data.

– Extraction of structured data: If you are searching for specific data on a website, you can use HTMLAgilityPack to sample and extract that data from the HTML code.

– Converting HTML to a structured format: If you receive an HTML document and need to convert it to a structured format for further processing or storage in a database, HTMLAgilityPack can be used to parse and create objects or data for further use.

HTMLAgilityPack helps to simplify the work with HTML and use the resulting data for further processing or storage in a database, which can be useful for obtaining information for further use in digitized data.

1. Basic technological stages

Conversion of electronic text of dictionaries into XML format

Converting electronic text of dictionaries into XML can be done by programming. However, this process is preceded by the extraction of all structural elements and the construction of the structure of a dictionary entry or text unit. Here is a general scheme of working with XML:

1. Reading the text. Read the electronic text of the dictionary according to the structure of its lexicogaphy system.

2. Parsing the text. Parsing the text into various structural elements of the dictionary, such as words, definitions, examples of usage, etc. To do this, you can use regular expressions or text processing libraries such as NLTK (Natural Language Toolkit) in Python.

3. Create an XML structure. For each selected element, create a corresponding XML element, for example, <word>, <definition>, <example>, etc. Add these elements to the XML tree. Develop a general schema.

4. Save in XML format. Save the resulting XML structure in an XML file.

5. Additional options. Add the ability to process different dictionary formats (for example, JSON, CSV). Add validation of the entered data to make sure that it corresponds to the expected format.

This work can be implemented using programming languages such as Python, Java, or any other language that supports text and XML processing.

Converting electronic text from dictionaries to XML can be useful for several reasons:

1. Structured data: XML allows you to structure information in the form of a tree, which makes it easier to work with data and use it later.

2. Interoperability with other systems: XML is a common format and is used to exchange data between different programs and systems.

3. Ease of processing and analysis: The XML structure makes it easy to process and analyze data using a variety of programs, including programming languages and other tools.

4. Support for different formats: XML can be used to represent different kinds of information, including texts, numbers, dates, and more, which can be useful for dictionary data.

5. Extensibility: XML makes it easy to add new elements and extend the data structure without significant changes to the existing code.

Consequently, converting dictionary data to XML can make it easier to process, share, and use in different applications and systems.

For example, let's show two dictionary entries in the structure of a lexicographic system and in XML format. (Dictionary of Ukrainian Biological Terminology)

Example 1

адвенти#вний (*рос.* адвенти#вный, *англ.* adventive) 1. Який розвивається не з ембріональних тканин точки росту, а із старіших частин рослини; **адвенти#вна бру#нька** *див.* **бру#нька**: **бру#нька адвенти#вна**; **адвенти#вна ембріоні#я** *див.* **ембріоні#я**: **ембріоні#я адвенти#вна**; **адвенти#вна поліембріоні#я** *див.* **поліембріоні#я: поліембріоні#я нуцеля#рна [адвенти#вна]**; **адвенти#вний за#родок** *див.* **за#родок: за#родок адвенти#вний**; **адвенти#вний о#рган** *див.* **о#рган: о#рган адвенти#вний**; **адвенти#вний па#гін** *див.* **па#гін: па#гін адвенти#вний** 2. Занесена людиною рослина в ту місцевість, де вона раніше не росла; **адвенти#вна росли#на** *див.* **росли#на: росли#на адвенти#вна**; **адвенти#вна вид** *див.* **вид: вид адвенти#вний.**

According to the structure of the dictionary article, we will highlight all the structural elements

ТБ [термінологічний блок]: **адвенти#вний** (*рос.* адвенти#вный, *англ.* adventive)
ТК_У [термінологічний комплекс український]: **адвенти#вний**
ЗТ [заголовний термін]: **адвенти#вний**

ТК_Р [термінологічний комплекс російський]: *рос.* адвенти#вный
ММ[маркер мови]: *рос.*
Т_Р [термін російський]:. адвенти#вный

ТК_А [термінологічний комплекс англійський]: *англ.* adventive
ММ [маркер мови]: *англ.*
Т_А [термін англійський]: adventive

СМБ [семантичний блок]: 1. Який розвивається не з ембріональних тканин точки росту, а із старіших частин рослини; **адвенти#вна бру#нька** *див.* **бру#нька**: **бру#нька адвенти#вна**; **адвенти#вна ембріоні#я** *див.*

ембріоні#я: **ембріоні#я адвенти#вна**; **адвенти#вна поліембріоні#я** *див.* **поліембріоні#я: поліембріоні#я нуцеля#рна [адвенти#вна]**; **адвенти#вний за#родок** *див.* **за#родок: за#родок адвенти#вний**; **адвенти#вний о#рган** *див.* **о#рган: о#рган адвенти#вний; адвенти#вний па#гін** *див.* **па#гін: па#гін адвенти#вний** 2. Занесена людиною рослина в ту місцевість, де вона раніше не росла; **адвенти#вна росли#на** *див.* **росли#на: росли#на адвенти#вна**; **адвенти#вна вид** *див.* **вид: вид адвенти#вний.**

$БТ_1$ [блок тлумачень]: 1. Який розвивається не з ембріональних тканин точки росту, а із старіших частин рослини;
НТ [номер тлумачення]: 1
$ТЛ_1$ [тлумачення]: Який розвивається не з ембріональних тканин точки росту, а із старіших частин рослини;

$БТ_2$ [блок тлумачень]: 2. Занесена людиною рослина в ту місцевість, де вона раніше не росла;
НТ [номер тлумачення]: 2
$ТЛ_2$ [тлумачення]: Занесена людиною рослина в ту місцевість, де вона раніше не росла;

$БП_1$ [блок посилань]: **адвенти#вна бру#нька** *див.* **бру#нька**: **бру#нька адвенти#вна**; **адвенти#вна ембріоні#я** *див.* **ембріоні#я**: **ембріоні#я адвенти#вна**; **адвенти#вна поліембріоні#я** *див.* **поліембріоні#я: поліембріоні#я нуцеля#рна [адвенти#вна]**; **адвенти#вний за#родок** *див.* **за#родок: за#родок адвенти#вний**; **адвенти#вний о#рган** *див.* **о#рган: о#рган адвенти#вний**; **адвенти#вний па#гін** *див.* **па#гін: па#гін адвенти#вний**
$ПБП_1$ [підблок посилань]: **адвенти#вна бру#нька** *див.* **бру#нька**: **бру#нька адвенти#вна**;
$ПБП_2$ [підблок посилань]: **адвенти#вна ембріоні#я** *див.* **ембріоні#я**: **ембріоні#я адвенти#вна**;
$ПБП_3$ [підблок посилань]: **адвенти#вна поліембріоні#я** *див.* **поліембріоні#я: поліембріоні#я нуцеля#рна [адвенти#вна]**;
$ПБП_4$ [підблок посилань]: **адвенти#вний за#родок** *див.* **за#родок: за#родок адвенти#вний**;
$ПБП_5$ [підблок посилань]: **адвенти#вний о#рган** *див.* **о#рган: о#рган адвенти#вний**;
$ПБП_6$ [підблок посилань]: **адвенти#вний па#гін** *див.* **па#гін: па#гін адвенти#вний**
$БП_2$ [блок посилань]: **адвенти#вна росли#на** *див.* **росли#на: росли#на адвенти#вна**; **адвенти#вна вид** *див.* **вид: вид адвенти#вний.**
$ПБП_1$ [підблок посилань]: **адвенти#вна росли#на** *див.* **росли#на: росли#на адвенти#вна**;
$ПБП_2$ [підблок посилань]: **адвенти#вна вид** *див.* **вид: вид адвенти#вний.**

Each sub-block consists of an addressee1, a link marker and an addressee2.

$ПБП_2$ [підблок посилань]: **адвенти#вна вид** *див.* **вид: вид адвенти#вний.**
САНТ [адресант]: **адвенти#вна вид**
МП[маркер посилань]: *див.*
САТ [адресат]: вид: вид адвенти#вний.

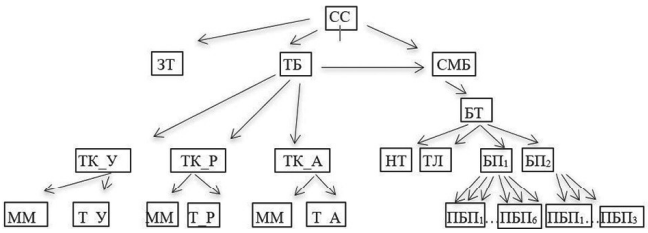

Figure 1: Outline of a dictionary article

Example 2

перетя#жка (*рос.* перетя#жка, *англ.* constriction) 1. Неспіралізована ділянка спіральної хромосоми, величина і локалізація якої варіюється; 2. (*англ.* strangulation) Дуже тонке, звужене місце; **перетя#жка втори#нна** (*рос.* перетя#жка втори#чная, *англ.* secondary chromosome strangulation) 1. Звужена частина, яка може локалізуватись у різних місцях хромосоми.
According to the structure of the dictionary article, we will highlight all the structural elements
$ТБ_1$ [термінологічний блок]: **перетя#жка** (*рос.* перетя#жка, *англ.* constriction
ТК_У [термінологічний комплекс український]: **перетя#жка**
ЗТ [заголовний термін]: **перетя#жка**
ТК_Р [термінологічний комплекс російський]: *рос.* перетя#жка
ММ[маркер мови]: *рос.*
Т_Р [термін російський]: перетя#жка
ТК_А [термінологічний комплекс англійський]: *англ.* constriction
ММ [маркер мови]: *англ.*
Т_А [термін англійський]: constriction

$ТБ_2$ [термінологічний блок]: 2. (*англ.* strangulation)
ТК_У [термінологічний комплекс український]: **перетя#жка**
ЗТ [заголовний термін]: **перетя#жка**
ТК_Р [термінологічний комплекс російський]: *рос.* перетя#жка
ММ[маркер мови]: *рос.*
Т_Р [термін російський]: перетя#жка
ТК_А [термінологічний комплекс англійський]: *англ.* strangulation
ММ [маркер мови]: *англ.*
Т_А [термін англійський]: strangulation

$СМБ_1$ [семантичний блок]: 1. Неспіралізована ділянка спіральної хромосоми, величина і локалізація якої варіюється;

БТ [блок тлумачень]: 1. Неспіралізована ділянка спіральної хромосоми, величина і локалізація якої варіюється;
НТ [номер тлумачення]: 1
ТЛ [тлумачення]: Неспіралізована ділянка спіральної хромосоми, величина і локалізація якої варіюється;
СМБ$_2$ [семантичний блок]: Дуже тонке, звужене місце; **перетя#жка втори#нна** (*рос.* перетя#жка втори#чная, *англ.* secondary chromosome strangulation) 1. Звужена частина, яка може локалізуватись у різних місцях хромосоми.
БТ [блок тлумачень]: Дуже тонке, звужене місце
НТ [номер тлумачення]: 2
ТЛ [тлумачення]: Дуже тонке, звужене місце
ПБТС [підблок словосполучень]: **перетя#жка втори#нна** (*рос.* перетя#жка втори#чная, *англ.* secondary chromosome strangulation) 1. Звужена частина, яка може локалізуватись у різних місцях хромосоми.
БТС [блок словосполучень]: **перетя#жка втори#нна** (*рос.* перетя#жка втори#чная, *англ.* secondary chromosome strangulation) 1. Звужена частина, яка може локалізуватись у різних місцях хромосоми.
ТБСЛ [термінологічний блок словосполучень]: **перетя#жка втори#нна** (*рос.* перетя#жка втори#чная, *англ.* secondary chromosome strangulation)
ТКС_У [термінологічний комплекс словосполучення укр.]: **перетя#жка втори#нна**
ТС_У [термінологічне словосполучення укр.]: **перетя#жка втори#нна**
ТКС_Р [термінологічний комплекс словосполучення рос.]: *рос.* перетя#жка втори#чная
ММ[маркер мови]: *рос.*
ТС_Р [термінологічне словосполучення рос.]: перетя#жка втори#чная
ТКС_А [термінологічний комплекс словосполучення англ.]: *англ.* secondary chromosome strangulation
ММ[маркер мови]: *англ.*
ТС_А [термінологічне словосполучення англ.]: secondary chromosome strangulation
БТСЛ [семантичний блок словосполучень]: 1. Звужена частина, яка може локалізуватись у різних місцях хромосоми.
НТСЛ [номер тлумачення словосполучення]: 1
ТЛС [тлумачення словосполучення]: Звужена частина, яка може локалізуватись у різних місцях хромосоми.

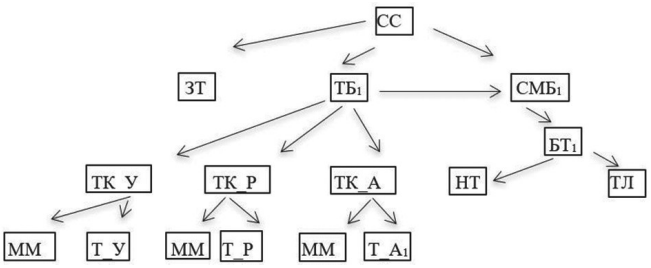

Figure 2.1: The scheme of articles is divided into two blocks, first

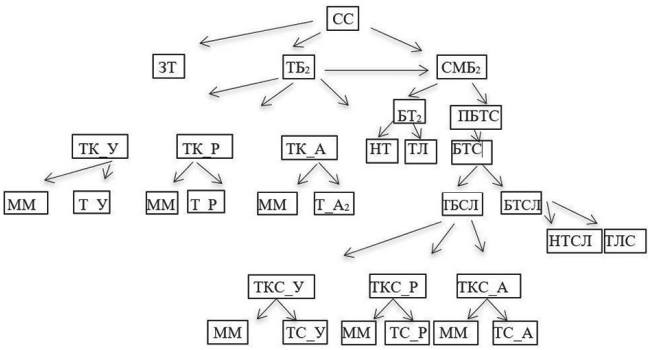

Figure 2.2: The scheme of articles is divided into two blocks, second

Example 1 to XML

```
<СС>
 <текст_СС><![CDATA[<B>адвенти#вний</B> (<I>рос.</I> адвенти#вный, <I>англ.</I> adventive) 1. Який розвивається не з ембріональних тканин точки росту, а із старіших частин рослини; <B>адвенти#вна бру#нька</B> <I>див.</I> <B>бру#нька: бру#нька адвенти#вна</B>; <B>адвенти#вна ембріоні#я</B> <I>див.</I> <B>ембріоні#я: ембріоні#я адвенти#вна</B>; <B>адвенти#вна поліембріоні#я</B> <I>див.</I> <B>поліембріоні#я: поліембріоні#я нуцеля#рна [адвенти#вна]</B>; <B>адвенти#вна за#родок</B> <I>див.</I> <B>за#родок: за#родок адвенти#вний</B>; <B>адвенти#вний о#рган</B> <I>див.</I> <B>о#рган: о#рган адвенти#вний</B>; <B>адвенти#вний па#гін</B> <I>див.</I> <B>па#гін: па#гін адвенти#вний</B> 2. Занесена людиною рослина в ту місцевість, де вона раніше не росла; <B>адвенти#вна росли#на</B> <I>див.</I> <B>росли#на: росли#на адвенти#вна</B>; <B>адвенти#вна вид</B> <I>див.</I> <B>вид: вид адвенти#вний</B>.]]></текст_СС>
  <ЗТ>адвенти#вний</ЗТ>
  <ТБ номер="1">
   <ТК_У номер="1">
    <Т_У>адвенти#вний</Т_У>
    <ММ>укр.</ММ>
   </ТК_У>
   <ТК_Р номер="1">
    <Т_Р>адвенти#вный</Т_Р>
    <ММ>рос.</ММ>
   </ТК_Р>
   <ТК_А номер="1">
    <Т_А>adventive</Т_А>
    <ММ>англ.</ММ>
   </ТК_А>
  </ТБ>
  <СМБ номер="1">
   <БТ номер="1">
    <НТ>1</НТ>
    <ТЛ>Який розвивається не з ембріональних тканин точки росту, а із старіших частин рослини</ТЛ>
```

```
    </БТ>
    <БТ номер="2">
     <НТ>2</НТ>
     <ТЛ>Занесена людиною рослина в ту місцевість, де вона раніше не росла</ТЛ>
    </БТ>
    <БП номер="1">
     <САНТ>адвенти#вна бру#нька</САНТ>
     <САТ>бру#нька: бру#нька адвенти#вна</САТ>
     <МП>див.</МП>
    </БП>
    <БП номер="2">
     <САНТ>адвенти#вна ембріоні#я</САНТ>
    <САТ>ембріоні#я: ембріоні#я адвенти#вна</САТ>
     <МП>див.</МП>
    </БП>
    <БП номер="3">
     <САНТ>адвенти#вна поліембріоні#я</САНТ>
    <САТ>поліембріоні#я: поліембріоні#я нуцеля#рна [адвенти#вна]</САТ>
     <МП>див.</МП>
    </БП>
    <БП номер="4">
     <САНТ>адвенти#вна за#родок</САНТ>
    <САТ>за#родок: за#родок адвенти#вний</САТ>
     <МП>див.</МП>
    </БП>
    <БП номер="5">
     <САНТ>адвенти#вний о#рган</САНТ>
     <САТ>о#рган: о#рган адвенти#вний</САТ>
     <МП>див.</МП>
    </БП>
    <БП номер="6">
     <САНТ>адвенти#вний па#гін</САНТ>
     <САТ>па#гін: па#гін адвенти#вний</САТ>
     <МП>див.</МП>
    </БП>
    <БП номер="7">
     <САНТ>адвенти#вна росли#на</САНТ>
    <САТ>росли#на: росли#на адвенти#вна</САТ>
     <МП>див.</МП>
    </БП>
    <БП номер="8">
     <САНТ>адвенти#вна вид</САНТ>
     <САТ>вид: вид адвенти#вний</САТ>
     <МП>див.</МП>
    </БП>
   </СМБ>
  </СС>
```

Example 2 to XML

```
<СС>
<текст_СС><![CDATA[<B>перетя#жка</B> (<I>рос.</I> перетя#жка, <I>англ.</I> constriction) 1. Неспіралізована ділянка спіральної хромосоми, величина і локалізація якої варіюється; 2. (<I>англ.</I> strangulation) дуже тонке, звужене місце; <B>перетя#жка втори#нна</B> (<I>рос.</I> перетя#жка втори#чная, <I>англ.</I> secondary chromosome strangulation) звужена частина, яка може локалізуватись у різних місцях хромосоми.]]></текст_СС>
  <ЗТ>перетя#жка</ЗТ>
  <ТБ номер="1">
   <ТК_У номер="1">
    <Т_У>перетя#жка</Т_У>
    <ММ>укр.</ММ>
   </ТК_У>
   <ТК_Р номер="1">
    <Т_Р>перетя#жка</Т_Р>
    <ММ>рос.</ММ>
   </ТК_Р>
   <ТК_А номер="1">
    <Т_А>constriction</Т_А>
    <ММ>англ.</ММ>
   </ТК_А>
  </ТБ>
  <ТБ номер="2">
   <ТК_У номер="1">
    <Т_У>перетя#жка</Т_У>
    <ММ>укр.</ММ>
   </ТК_У>
   <ТК_Р номер="1">
    <Т_Р>перетя#жка</Т_Р>
    <ММ>рос.</ММ>
   </ТК_Р>
   <ТК_А номер="1">
    <Т_А>strangulation</Т_А>
    <ММ>англ.</ММ>
   </ТК_А>
  </ТБ>
  <СМБ номер="1">
   <БТ номер="1">
    <НТ>1</НТ>
    <ТЛ>дуже тонке, звужене місце;</ТЛ>
   </БТ>
   <БТС номер="1">
    <ТБСЛ номер="1">
     <ТКС_У номер="1">
      <ТС_У>перетя#жка втори#нна</ТС_У>
      <ММ>укр.</ММ>
     </ТКС_У>
     <ТКС_Р номер="1">
      <ТС_Р>перетя#жка втори#чная</ТС_Р>
      <ММ>рос.</ММ>
     </ТКС_Р>
     <ТКС_А номер="1">
      <ТС_А>secondary chromosome strangulation</ТС_А>
```

```
        <ММ>англ.</ММ>
       </ТКС_А>
      </ТБСЛ>
      <БТСЛ номер="1">
       <НТСЛ>1</НТСЛ>
      <ТЛС>звужена частина, яка може локалізуватись
у різних місцях хромосоми</ТЛС>
      </БТСЛ>
     </БТС>
    </СМБ>
    <СМБ номер="2">
     <БТ номер="1">
      <НТ>1</НТ>
      <ТЛ>Неспіралізована ділянка спіральної хромо-
соми, величина і локалізація якої варіюється</ТЛ>
     </БТ>
    </СМБ>
   </СС>
   <СС>
```

2. Stages of database formation

Creating a database (DB) from an XML file can be done in several steps:

Creating a database schema. Define the data structure that needs to be stored in the database. The structure of the XML file is taken into account and the tables and fields for displaying this data are determined.

Creating tables in the database. Using SQL or another query language, tables are created in the database that correspond to the data structure from the XML file.

Loading data from XML. Tools or programming languages that support working with XML are used (for example, Python with the ElementTree library or .NET-based programming languages using XmlDocument). The XML file is parsed and the data is inserted into the database schema.

Using the HTMLAgilityPack, the entire input text is divided into articles and structural elements are extracted from them (stems).

Examples (Dictionary Lexicon of Polish and Ukrainian Active Phraseology):

```
HtmlDocument HtmlDoc = new HtmlDocument();
1. Load the text from the file HtmlDoc.
LoadHtml(input);
```

Selecting articles HtmlNodeCollection Nodes = HtmlDoc.DocumentNode.SelectNodes("//article");

After each selection of structural elements, perform this check

```
if (Nodes != null)
{
foreach (HtmlNode S_Node in Nodes)// Each article
HtmlNode tmp_node = S_Node.
SelectSingleNode(".//p[@class = 'j']"); // selecting one
```

element from the article by type <p class="t">. The dot here means that we are "looking" at only one article

```
String text = tmp_node.InnerHtml; // Get the text
that is in (between) the tags
Int ID_Ukr = tmp_node.Attributes["href"].Value; //
We got the value of the attribute of type <... href="***"
...>
```

2. Transferring the received data to an instance of the class. With a check whether there is already a "reverse" article (1 copy for both pol-UKR and ukr-pol. We look to see if there is, for example, the same case among the processed ones and, if so, just add the ID and letter in the language) ATTENTION the class should have a separate one (for the database), like this:

```
[BsonId]
public int ID { get; set; }
```

3. Loading data into the database. Examples:

```
private LiteDatabase DB { get; set; }
private LiteCollection<ReestrUnitMember> R_
Members { get; set; }
private string DB_Path { get; set; }

DB_Path = path; // Name and (optionally) path of
the database file
DB = new LiteDatabase(path);
R_Members = DB.GetCollection<ReestrUnitMemb
er>("r_members"); // ReestrUnitMember - class name,
r_members - collection name
R_Members.EnsureIndex("Inner_ID", "$.Inner_
ID"); // Create an index. The 1st is the name of the in-
dex, the 2nd is the name of the parameter in the class, $.
is required
R_Members.EnsureIndex("Orig_Inner_ID", "$.Orig_
Inner_ID");
R_Members.Insert(Member); // Add a new instance
of the class (article) to the database.
```

Automation of the process. Consider automating this process, especially if you have a large number of XML files or if you plan to update data on a regular basis.

Verification and optimization. You need to make sure that the data is loaded correctly into the database and perform optimization, if necessary, such as using indexes to improve query performance.

The general conclusion is that the formation of a database from an XML file includes several key stages, from creating schema and tables to loading data and its further processing in the database, using appropriate tools and libraries.

3. Database protection

In today's digitalized society, an important criterion is the use of modern methods and means of protecting information, as any digitized data can be compromised or stolen. One of the most reliable methods of protection is the use of quantum key distribution protocols, one of which is Twin field (Table 1). The use of the Twin Field protocol to protect LiteDB involves the use of technology

based on the properties of a dual field of quantum states to create a secure key exchange channel. This allows you to achieve a high level of protection in the exchange of keys between the sender and the recipient in a quantum environment.

Table 1.

Principles of the TFmod quantum key distribution protocol

Criteria	Uncertainty principle	Measurement independence principle	Information transfer fidelity principle
1. Using quantum properties	Takes advantage of quantum uncertainty	Applies independent measurements	Guarantees flawless photon transmission
2. Security of key transfer	Uses quantum properties to protect keys	Ensures unconditional security of key transfer	Applies interception detection mechanisms
3. Interception detection	Detects any attempted interception	Detects any unauthorized interference	Provides detection of any changes or distortions
4. Measurement independence	The measurement is carried out independently	Ensures independence between measurements	Uses independent channels for photon transmission
5. Ensuring transmission accuracy	Uses error correction mechanisms	Provides high transmission accuracy	Applies error correction methods
6. Efficiency and speed	Uses optimized algorithms	Provides high transmission speed	Highly productive and efficient
7. Scalability	Implemented for large networks	Scalable for extended networks	Applies to different network scales

The essence of Twin Field is to use two parameters (polarization and time) to encode and transmit quantum bits (qubits). This provides a double level of information that can be used to create a cryptographically secure key. One of the important advantages of the protocol is its ability to detect key interception attempts. According to the principles of quantum mechanics, any attempt to measure a quantum state leads to its change, which is automatically detected by the system, ensuring the security of key exchange. In this context, the use of Twin Field to protect LiteDB includes the creation of cryptographically secure keys that are used to encrypt and decrypt the database. This approach helps to ensure a high level of protection and prevents attacks on encryption keys in a quantum environment.

To ensure the process of applying the Twin Field protocol to protect LiteDB, the following steps should be followed:

1. Quantum key exchange: A quantum key exchange process is performed between the sender and the receiver using Twin Field. During this exchange, the double field must be considered to provide an additional layer of information and security.

2. Interception detection: Using a property of quantum mechanics, such as the non-determinism of measurements, to detect any attempts to intercept keys. This can be realized by using quantum properties that change with any observation.

3. Encrypting and decrypting data in LiteDB: using the obtained quantum keys to encode and decode data. This will ensure the confidentiality and integrity of information in the database.

4. Ongoing monitoring and support of the system: monitoring the state of the system and detecting any anomalies or attack attempts in time. Ensure that keys are regularly updated and the system is improved to reflect the latest advances in quantum security.

When considering the integration of a quantum key distribution device on the same server as the LiteDB database, the key aspect is to ensure the comprehensive protection and efficiency of this infrastructure. For this purpose, it is important to address physical, technical, and network aspects. In terms of physical protection, you should place the device in a secure room using restricted access mechanisms.

Efficient use of server resources should include optimizing the configuration for quantum signal processing. It is also important to consider network aspects, using modern encryption tools and firewalls to protect against external attacks. Maintaining the continuous operation of the system includes the implementation of a monitoring and maintenance system to detect possible malfunctions and anomalies. The use of logging and auditing methods allows you to maintain event logs and identify potential security threats.

The level of modern security bypass systems should also be taken into account. level of modern security bypass systems should also be taken into account. Therefore, the standard Twin Field protocol should be enhanced with additional methods to ensure resilience. One of the best ways is to improve the security system using quantum identification and quantum state multiplexing methods, which includes several promising aspects. First of all, there is a high resistance to cyber threats due to the application of quantum identification principles to generate cryptographic keys. This is important because this approach makes it difficult to crack encryption algorithms based on classical computing. The second aspect is the use of quantum state multiplexing, which allows more information to be transmitted simultaneously over communication channels. This can lead to improved performance and efficient processing of large amounts of data. The third aspect involves the use of quantum states to transmit

keys, reducing the likelihood of interception. Observing a quantum state leads to a change in the state itself, which makes detecting attacks and hacking even more difficult. In addition, the use of quantum methods can increase the level of confidentiality and data security, as they are based on the physical principles of quantum mechanics. This approach makes the keys unique and unpredictable, which makes the system more relevant to modern cyber-security challenges.

Conclusion

In this article, we have considered important aspects of forming digitized data in the LiteDB database, using the HTMLAgilityPack library to efficiently process HTML in the .NET environment and ensure the reliability of this system through the use of quantum cryptography.

Digitization of data is becoming an integral part of modern society, and using LiteDB provides us with a simple and effective solution for storing this data. Its embeddability, JSON/BSON data format, ease of use, and scalability make it an excellent choice for projects of all sizes.

The HTMLAgilityPack library allows you to efficiently process HTML pages in the .NET environment, making it an ideal tool for extracting structured information from web resources.

Special attention is paid to the use of quantum cryptography to ensure system security. Its high resistance to various cyber threats, even under conditions of uncertainty, makes it an advanced tool for protecting confidential information. An important aspect is the use of the Twin field protocol, which, due to its resistance to attacks under conditions of uncertainty, forms a reliable foundation for data security.

References

[1] *Grodzinsky, D. M., Simonenko, L. O. et al..* (2012). Ukrainian biological terminology Dictionary. Kyiv: KMM, 2012.

[2] Linguistic and information studies: works of the Ukrainian language and information fund of the NAS of Ukraine: in 5 volumes. (2018). V. A. Shyrokov et al. Vol. 1: Kyiv.

[3] *Ben Bongalon, Joel Ilao, Ethel Ong, Rochelle Irene Lucas, Melvin Jabar.* (2021). Using Open-Source Tools to Digitise Lexical Resources for Low-Resource Languages. Proceedings of the eLex 2021 conference. 5–7 July 2021, virtual. Brno: Lexical Computing CZ, s.r.o.

[4] *Shyrokov, V. A. (Ed.)* (2011). Computer lexicography. Kyiv: Naukova dumka

[5] *Karpova, O.* (2009). Lexicography and Terminology: A Worldwide Outlook / Olga Karpova, Faina Kartashkova. Cambridge: Cambridge Scholars Publishing.

[6] *Shyrokov V. A. etc.* (2018). Linguistic and information studies: works of the Ukrainian Language and Information Fund NAS of Ukraine: in 5 vols. Vol. 5: Virtualization of linguistic technologies. Kyiv: Ukrainian Lingua-Information Fund of NAS of Ukraine. URL: https://movoznavstvo.org.ua/files/Ling_inf_studio_TOM_5_umif_B5.pdf doi: 10.33190/978-966-02-8683-2/8690-0

[7] *Baiisa, V, Blahuš, M., Cukr, M., Herman, O., et al.* (2019). Automating Dictionary Production: a Tagalog-English-Korean Dictionary from Scratch. Proceedings of the eLex 2019 conference. 1-3 October 2019, Sintra, Portugal. Brno: Lexical Computing CZ, s.r.o.

[8] *Shyrokov V. A.* (2018). Grammatical systems: phenomenological approach. V. A. Shyrokov, T. P. Lyubchenko, I. V. Shevchenko, K. V. Shyrokov. Kyiv: Naukova dumka.

[9] *Gnatyuk, S., Okhrimenko, T., Dorozhynskyy, S., Fesenko, A,.* (2019). Review of modern quantum key distribution protocols, Scientific and Practical Cyber Security Journal (SPCSJ), 4(1), 56–60. ISSN 2587-4667.

[10] *Buck, L. E., Bodenheimer, B.* (2021). Privacy and Personal Space: Addressing Interactions and Interaction Data as a Privacy Concern. 2021 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW), Lisbon, Portugal, 399-400, doi: 10.1109/VRW52623.2021.00086.

[11] *Kernerman, I.* (2015). A multilingual trilogy: Developing three multi-language lexicographic datasets. Electronic Lexicography in the 21st Century: Linking lexical data in the digital age. Proceedings of eLex 2015, 11-13 August 2015, 372-383. URL: https://elex.link/elex2015/

[12] *Liu, X., Xie, J., Kang, J., Zhang, M.* (2022). Twin-Field Quantum Key Distribution Protocol with Heralded Single-Photon Source, 2022 IEEE 12th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing, China, 94-97, doi: 10.1109/ICEIEC54567.2022.9835076.

The article was delivered to editorial stuff on the 17.01.2025