УДК 33 **DOI: 10.34670/AR.2022.88.73.023**

Об алгоритмической и интеллектуальной обработке данных: экономические аспекты

Костиков Юрий Александрович

Кандидат физико-математических наук,
заведующий кафедрой 812,
Московский авиационный институт
(национальный исследовательский университет),
125993, Российская Федерация, Москва, Волоколамское шоссе, 4;
e-mail: jkostikov@mail.ru

Романенков Александр Михайлович

Кандидат технических наук,
доцент кафедры 812,
Московский авиационный институт
(национальный исследовательский университет),
125993, Российская Федерация, Москва, Волоколамское шоссе, 4;
e-mail: romanaleks@gmail.com

Скачков Артем Сергеевич

Инженер,
Московский авиационный институт
(национальный исследовательский университет),
125993, Российская Федерация, Москва, Волоколамское шоссе, 4;
емейл scarlett1965@inbox.ru

Аннотация

Статья посвящена функциональной структуре, алгоритмам работы и программной реализации информационного модуля для согласования терминов и их синонимов при проведении распределенных исследований экономических и социальных процессов. Предполагается, что исследования проводятся многочисленными независимыми экспертами, находящимися в различных городах и связанными с единым центром обработки и анализа данных локальными и глобальными сетями. Реализация требуемого функционала выполняется с использованием современных средств разработки программного обеспечения. Предложены функциональные и структурные схемы программного продукта. Представлены технологическая цепочка согласования параметров и элементы программного кода, которые реализуют обработку советующих данных.

Для цитирования в научных исследованиях

Костиков Ю.А., Романенков А.М., Скачков А.С. Об алгоритмической и интеллектуальной обработке данных. Программно-аналитический комплекс согласования терминов // Экономика: вчера, сегодня, завтра. 2022. Том 12. № 4А. С. 232-242. DOI: 10.34670/AR.2022.88.73.023

Ключевые слова

Согласование терминов, словарь синонимов, алгоритмическая и интеллектуальная обработка данных, программно-аналитический комплекс.

Введение

В настоящее время проводится большое количество различных аналитических, статистических, социальных, медицинских, психологических и других исследований, в которых участвует большое количество людей и организаций, использующих близкую, но не идентичную терминологию. Отсутствие единообразия в терминологии является характерной чертой подобных исследований, так как процесс согласования единой системы терминов между всеми участниками существенно замедлил бы их проведение, а порой сделал бы невозможным. В то же время их результаты, будучи обработаны в едином аналитическом центре, представляют безусловный интерес с точки зрения как научных, так и практических приложений. Важной частью работы единого аналитического центра является выработка и согласование унифицированной терминологии исследования.

Статья посвящена методике динамического согласования терминологии в таких исследованиях, включающей в себя построение в общем аналитическом центре единого терминологического словаря, состоящего из эталонных терминов и их синонимов. Согласование словаря осуществляется дистанционно по локальным или глобальным сетям с помощью разработанного программно-аналитического комплекса. Описанный функционал может использоваться в программном обеспечении для обработки данных эксперимента [Костиков, Павлов, Романенков, 2018].

Функциональная и структурная схемы программно-аналитического комплекса

Схема функционального взаимодействия локальных центров исследований с главным аналитическим центром представлена на рисунке 1.

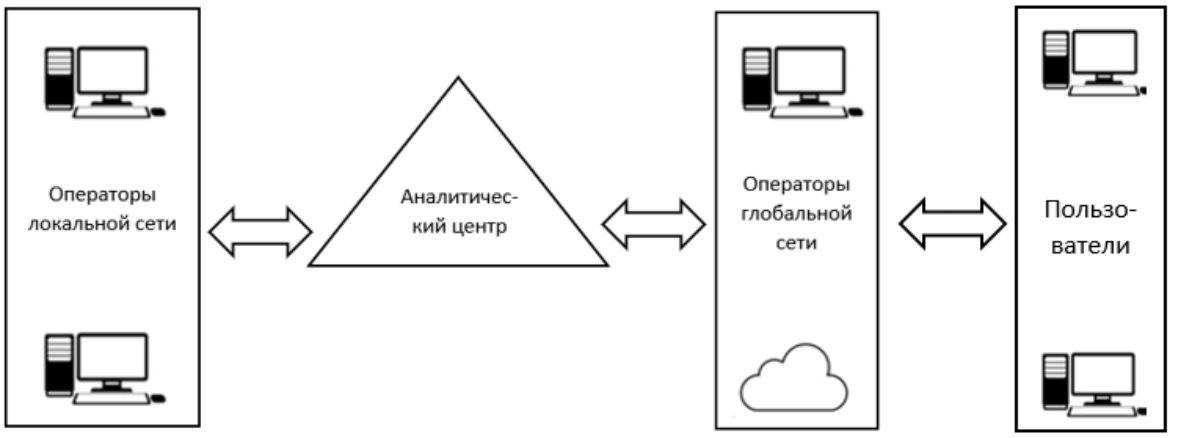

Рисунок 1 - Функциональная схема взаимодействия локальных центров исследований с главным аналитическим центром

Схема структуры программно-аналитического комплекса обработки результатов представлена на рисунке 2.

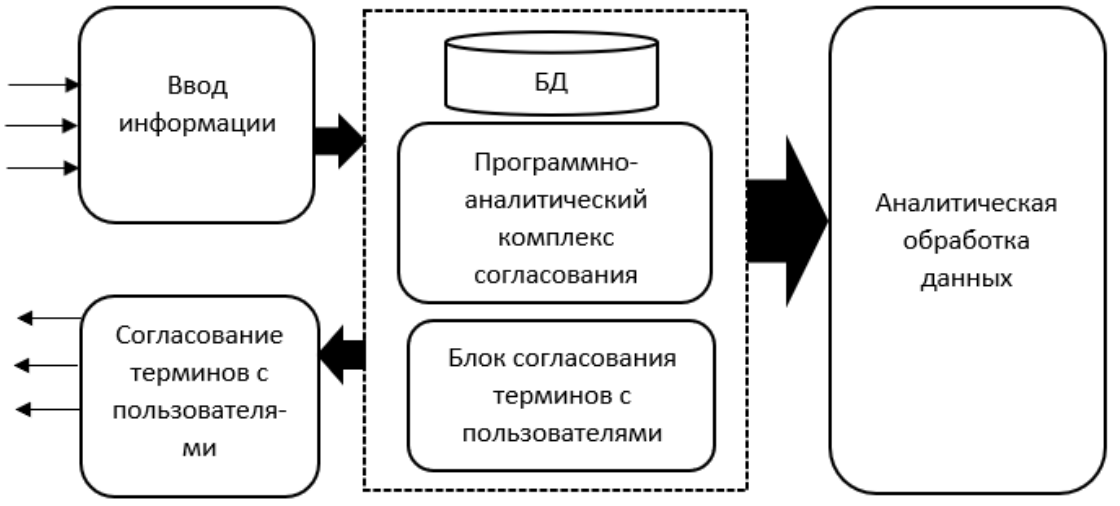

Рисунок 2 - Структурная схема программно-аналитического комплекса обработки результатов

Работа посвящена алгоритмам работы программно-аналитического комплекса согласования терминов.

Словарь терминов

Словарь терминов представляет из себя базу данных с управляющим ею комплексом, осуществляющим введение новых терминов (эталонов), синонимов и управляющим их согласованием с локальными исследовательскими центрами. База данных содержит словарь эталонных терминов и их синонимов, которые представлены в таблице parameters (таблица 1). Разработка структуры хранения данных и методов взаимодействия с изолированным хранилищем из клиентского приложения опирается на стандартные паттерны проектирования [Костиков, Павлов, Романенков, 2017; Гамма, Хелм, Джонсон, Влиссидес, 2016].

Таблица 1 - Структура таблицы parameters

Поле	Тип	Описание
id	integer	ID термина
name	varchar	Название термина
confirmed	boolean	Подтвержден ли термин – прошел ли проверку оператором
standard_id	integer	ID вышестоящего термина – эталона (если термин является синонимом, иначе – NULL)

Таблица parameters организована так, что с ее помощью и без вспомогательных таблиц можно восстановить все дерево терминов. Это экономит место в базе данных и время выполнения запросов.

Каждая запись в таблице – отдельный термин и отдельная ветка в дереве словаря. В записи хранятся следующие характеристики: название термина name, флаг подтвержденности термина

confirmed и поле со ссылкой на родительский элемент в дереве терминов standard_id.

Флаг confirmed отвечает за то, обработан ли в данный момент термин в текущей записи; если термин не обработан, то он будет отображаться у оператора в списке «неподтвержденных» терминов. После подтверждения термин переместится в список «подтвержденные».

Поле standard_id хранит идентификатор id эталонного термина из таблицы parameter (поле ссылается на другое поле в этой же таблице). Если термин в текущей записи является эталоном, то поле standard_id будет «пустым» – в нем будет лежать значение «NULL», а если, наоборот, является синонимом, то в поле standard_id будет лежать id эталонного термина. Во время обработки данных исследований синонимичные термины будут заменяться на эталонные названия терминов, чьи id лежат в полях standard_id.

Алгоритмы формирования единого дерева терминов и их синонимов

Перед добавлением в дерево элемент проходит обработку оператором, которая обычно занимает от одного до нескольких этапов.

1. Если поступивший термин является эталоном, то он подтверждается как «эталонный» и добавляется в корень дерева (рис. 3-4).

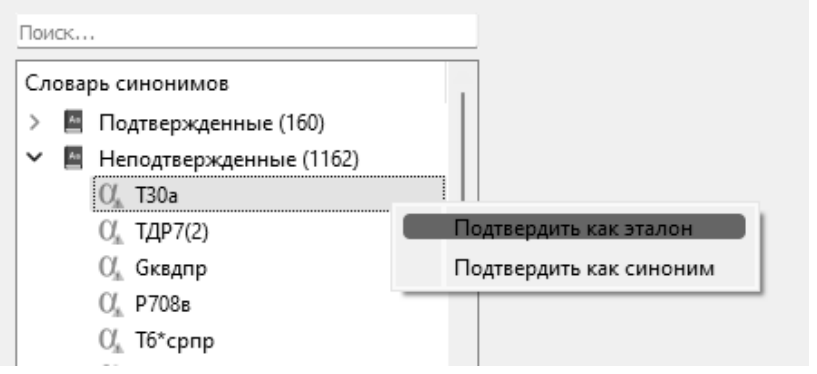

Рисунок 3 - Контекстное меню неподтвержденного термина

Выбор пункта меню «Подтвердить как эталон»

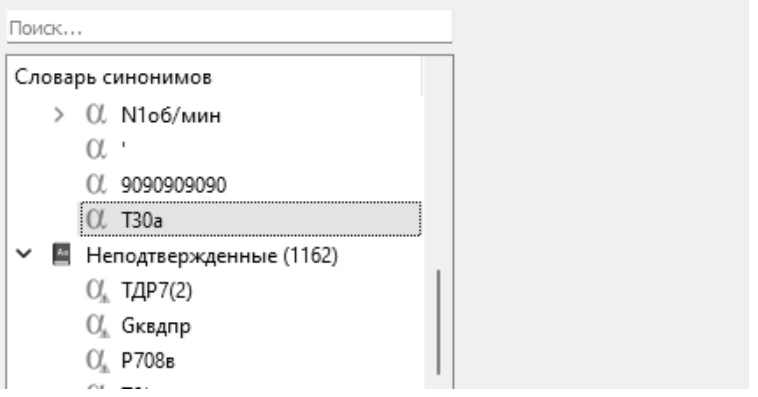

Рисунок 4 - Результат выбора пункта «Подтвердить как эталон». Перемещение неподтвержденного термина в список эталонов

2. Если поступивший термин является синонимом к существующему в дереве эталонному термину, то он подтверждается как «синоним к эталонному» и закрепляется в дереве дочерней веткой к термину-эталону (рис. 5-6).

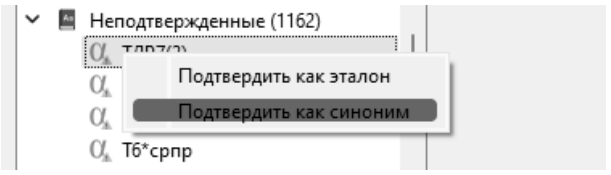

Рисунок 5- Контекстное меню неподтвержденного параметра

Выбор пункта меню «Подтвердить как синоним»

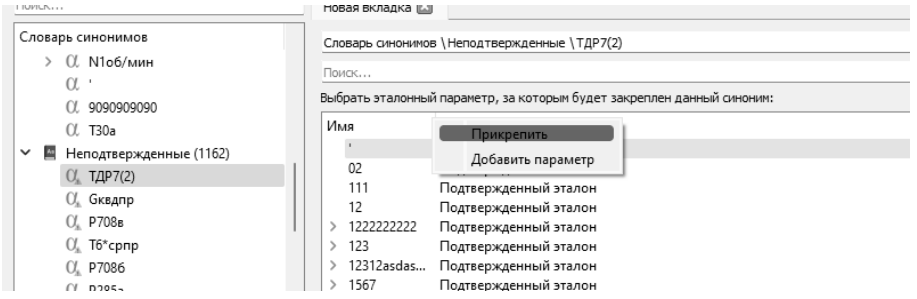

Рисунок 6 - Выбор эталонного параметра. Контекстное меню эталона

Выбор пункта меню «Прикрепить»

3. Если поступивший термин является синонимом к несуществующему в дереве эталонному термину, то сначала в корне дерева создается новый термин-эталон, а затем к нему привязывается поступивший синоним (рис. 7-8).

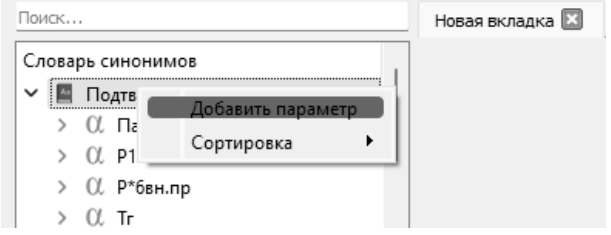

Рисунок 7 - Контекстное меню подтвержденного параметра

Выбор пункта меню «Добавить параметр»

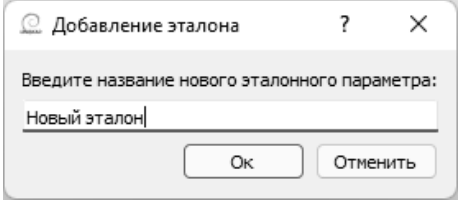

Рисунок 8 - Диалог добавления нового эталонного параметра

4. Если поступивший термин более похож на эталон, чем уже закрепленный в дереве синонимичный эталон, то происходит «перестановка»: термин-эталон открепляется от дерева, поступивший термин подтверждается как эталон, после чего открепленный термин закрепляется за новым добавленным (рис. 9).

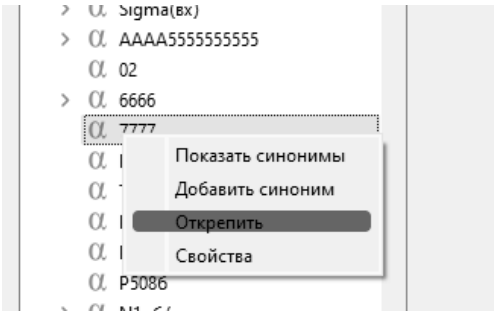

Рисунок 9 - Контекстное меню подтвержденного параметра

Выбор пункта меню «Открепить»

Программная реализация и применения

Программный комплекс разработан на языке Python 3.9 [SQLAlchemy, www], интерфейс программ построен с помощью графической библиотеки PySide2 [Fitzpatrick, 2019; Прохоренок, 2012].

Дерево элементов описано как связка представление + модель с помощью классов «PySide2.QtWidgtes.QTreeView» [QTreeView, www] (листинг 1) и «PySide2.QtWidgets.QAbstractItemModel» [QAbstractItemModel, www] (листинг 2).

```
class TreeView(QtWidgets.QTreeView):
 def __init__(self, parent=None):
 super().__init__(parent).
 self.setContextMenuPolicy(Qt.CustomContextMenu)
 self.setEditTriggers(QAbstractItemView.SelectedClicked)
 self.__create_connections()
 self.setSelectionMode(QAbstractItemView.ExtendedSelection)
 self.setExpandsOnDoubleClick(False)
 self.header().setSectionsClickable(True)
```

Листинг 1 - Переопределение класса QTreeView.

```
class ParameterTreeModel(TreeModel):
 HEADER = "Словарь синонимов"
 def __init__(self, groups: ty.List[ParameterGroup], parent=None):
 super().__init__(None, parent)
 self._root = RootItem(parent=self)
 self._root.set_children([ParameterGroupItem(group, self._root) for group in groups])
```

Листинг 2 - Переопределение модели QAbstractItemModel для TreeView

Контекстное меню элементов дерева терминов реализовано путем создания экземпляра класса «PySide2.QtWidgtes.QMenu» [QMenu, www] и наполнения его элементами меню

«PySide2.QtWidgets.QActions» [QAction, www] (листинг 3).

```
 menu = QMenu(self._main_window)
 source_current = self.source_index(current)
 if not current.isValid():
 return menu
 item = source_current.internalPointer()
 if isinstance(item, ParameterGroupItem):
 if item.name == 'Подтвержденные':
 menu.addAction(
 QAction(
 _("Add parameter"),
 triggered=lambda: self._model.add_parameter(self._main_window, source_current),
 parent=menu,
 )
 )
 sort_menu = QMenu(
 _("Sort"),
 parent=menu,
 )
 menu.addMenu(sort_menu)
 sort_by_name_action = QAction(
 _("Sort by name"),
 triggered=lambda: self.sort('name'),
 parent=menu,
 )
 sort_by_date_action = QAction(
 _("Sort by date"),
 triggered=lambda: self.sort('join_datetime'),
 parent=menu,
 )
 sort_menu.addAction(sort_by_name_action)
 sort_menu.addAction(sort_by_date_action)

 if isinstance(item, ParameterItem):
 parameter = item.item_data
 if parameter.confirmed:
 if parameter.standard_id:
 menu.addAction(
 QAction(_("Unfix"), triggered=lambda: self._model.unconfirm_parameter(source_current),
 parent=menu))
 menu.addAction(
 QAction(_("Properties"), triggered=lambda: self.properties(source_current), parent=menu))
 else:
 menu.addAction(QAction(_("View synonyms"), triggered=lambda: self.open_item(current),
parent=menu))
```

```
menu.addAction(
QAction(_("Add synonym"),
triggered=lambda: self._model.add_parameter(self._main_window, source_current),
parent=menu))
if not source_current.internalPointer().has_children:
menu.addAction(
QAction(_("Unfix"), triggered=lambda: self._model.unconfirm_parameter(source_current),
parent=menu))
menu.addAction(
QAction(_("Properties"), triggered=lambda: self.properties(source_current), parent=menu))
else:
menu.addAction(
QAction(_("Confirm as standard"), triggered=lambda:
self._model.confirm_as_standard(source_current),
parent=menu))
menu.addAction(QAction(_("Confirm as synonym"), triggered=lambda:
self.open_item(current), parent=menu))
return menu
```

Листинг 3 - Создание меню

В качестве хранилища данных используется база данных PostgreSQL 9.6 [SQLAlchemy, www] Весь функционал работы дерева перенесен в хранимые процедуры, потому что процедуры компилируются на стороне БД и позволяют быстрее выполнять запросы, что ускоряет процесс работы всего программного комплекса. Взаимодействие с базой данных организовано посредством библиотеки sqlalchemy [Psycopg – PostgreSQL database adapter for Python, www], которая, в свою очередь, использует psycopg2 [Overwiew, www] как «движок» для работы с БД. Процедура для вызова хранимых процедур представлена в листинге 4.

```
def stored_procedure(self, default: "Any" = None, isolated: bool = False, modifying: bool = False):
def wrapper(function: Callable):
sig = signature(function)
return_type = sig.return_annotation

@wraps(function)
async def inner(*args, **kwargs):
async with self._transaction(isolated) as transaction:
try:
res = await function(*args, **kwargs)
except DBAPIError as ex:
await transaction.rollback()
logging.error(f"Ошибка выполнения хранимой процедуры: {ex}.")
return default
if modifying:
self._dirty = True
return parse_obj_as(return_type, res)
```

```
 return inner
 return wrapper
```

Листинг 4 - Описание класса Session – подключение к БД и выполнение запросов

Приходящая в качестве результата выполнения запросов информация обрабатывается с помощью библиотеки pydantic (листинг 5). Каждая строка запроса «парсится» в отдельные объекты классов, наследуемых от «pydantic.BaseModel», что упрощает работу с результатами запросов.

```
class BaseSchema(BaseModel):
 class Config:
 allow_population_by_field_name = True
class ParameterGroup(BaseSchema):
 name: str = Field(..., alias="parameter_group", title="Название группы параметров")
 count: int = Field(..., title="Количество параметров в группе")
 confirmed: bool = Field(..., title="Состояние параметров в группе (подтвержден/нет)")
class DictParameter(BaseSchema):
 id: int = Field(..., title="Идентификатор параметра")
 name: str = Field(..., title="Название параметра")
 confirmed: bool = Field(..., title="Состояние параметра (подтвержден/нет)")
 standard_id: Optional[int] = Field(..., title="Идентификатор эталона")
 join_datetime: datetime = Field(..., title="Дата первого появления параметра")
 has_synonyms: bool = Field(False, title="Имеются ли синонимы у параметра")
```

Листинг 5 - Использование pydantic для «парсинга» информации из запросов

Заключение

Таким образом, в статье рассмотрен метод разработки программного модуля для согласования терминов. Фактически предложен алгоритм установления и поддержания ассоциативной связи между ключевым термином и множеством синонимов для него, которых может быть большое количество. Разработана специализированная табличная структура для реализации предложенного алгоритма в реляционных СУБД, описан функционал интерфейса пользовательского приложения, предназначенного для согласования параметров. Представлены примеры программного кода, которые реализуют необходимый функционал.

Библиография

1. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Д. Приемы объектно-ориентированного проектирования. Паттерны проектирования. СПб.: Питер, 2016.
2. Костиков Ю.А., Павлов В.Ю., Романенков А.М. Применение современных технологий, подходов и шаблонов проектирования при реализации масштабируемого клиент-серверного программного комплекса // Современная наука: актуальные проблемы теории и практики. Серия: Естественные и технические науки. 2017. № 12/2. С. 37-44.
3. Костиков Ю.А., Павлов В.Ю., Романенков А.М. Программный комплекс для моделирования данных натурного эксперимента // Перспективы науки. 2018. № 7 (106). С. 39-43.

4. Прохоренок Н.А. Python 3 и PyQT5. Разработка приложений. СПб: БХВ-Петербург, 2012. 704 с.
5. Шёниг Ганс-Юрген. PostgreSQL 11 // Мастерство разработки. 2018.
6. Fitzpatrick M. Create GUI Applications with Python & Qt5. 2019. 791 p.
7. SQLAlchemy. URL: https://docs.python.org/3.9/tutorial.
8. Overwiew. URL: https://pydantic-docs.helpmanual.io.
9. Psycopg – PostgreSQL database adapter for Python. URL: https://www.psycopg.org/docs.
10. QAbstractItemModel. URL: https://pyside.github.io/docs/pyside/PySide/QtCore/QAbstractItemModel.html.
11. QAction. URL: https://pyside.github.io/docs/pyside/PySide/QtGui/QAction.html
12. QMenu. URL: https://pyside.github.io/docs/pyside/PySide/QtGui/QMenu.html
13. QTreeView. URL: https://pyside.github.io/docs/pyside/PySide/QtGui/QTreeView.html.

On algorithmic and intelligent data processing. Software-analytical complex for harmonization of terms

Yurii A. Kostikov

PhD in Physical and Mathematical Sciences,
Head of the Department 812,
Moscow Aviation Institute (National Research University),
125993, 4 Volokolamskoe highway, Moscow, Russian Federation;
e-mail: jkostikov@mail.ru

Aleksandr M. Romanenkov

PhD in Technical Sciences, Associate Professor,
Department 812,
Moscow Aviation Institute (National Research University),
125993, 4 Volokolamskoe highway, Moscow, Russian Federation;
e-mail: romanaleks@gmail.com

Artem S. Skachkov

Engineer,
Moscow Aviation Institute (National Research University),
125993, 4 Volokolamskoe highway, Moscow, Russian Federation;
e-mail: scarlett1965@inbox.ru

Abstract

The article is devoted to the functional structure, operation algorithms and software implementation of the information module for the harmonization of terms and their synonyms when conducting distributed studies of economic and social processes. It is assumed that the research is carried out by numerous independent experts located in different cities and connected to a single data processing and analysis center by local and global networks. The implementation of the required functionality is carried out using modern software development tools. The authors propose functional and structural diagrams of the software product. The technological chain of parameter matching and program code elements that implement the processing of advising data are presented.

For citation

Kostikov Yu.A., Romanenkov A.M., Skachkov A.S. (2022) Ob algoritmicheskoi i intellektual'noi obrabotke dannykh. Programmno-analiticheskii kompleks soglasovaniya terminov [On algorithmic and intelligent data processing. Software-analytical complex for harmonization of terms]. *Ekonomika: vchera, segodnya, zavtra* [Economics: Yesterday, Today and Tomorrow], 12 (4A), pp. 232-242. DOI: 10.34670/AR.2022.88.73.023

Keywords

Harmonization of terms, dictionary of synonyms, algorithmic and intelligent data processing, software-analytical complex.

References

1. Fitzpatrick M. (2019) Create GUI Applications with Python & Qt5.
2. Gamma E., Khelm R., Dzhonson R., Vlissides D. (2016) Priemy ob"ektno-orientirovannogo proektirovaniya. Patterny proektirovaniya [Techniques of object-oriented design. Design patterns]. Saint Petersburg: Piter Publ.
3. Kostikov Yu.A., Pavlov V.Yu., Romanenkov A.M. (2017) Primenenie sovremennykh tekhnologii, podkhodov i shablonov proektirovaniya pri realizatsii masshtabiruemogo klient-servernogo programmnogo kompleksa [Application of modern technologies, approaches and design patterns in the implementation of a scalable client-server software package]. Sovremennaya nauka: aktual'nye problemy teorii i praktiki. Seriya: Estestvennye i tekhnicheskie nauki [Modern Science: Actual Problems of Theory and Practice. Series: Natural and technical sciences], 12/2, pp. 37-44.
4. Kostikov Yu.A., Pavlov V.Yu., Romanenkov A.M. (2018) Programmnyi kompleks dlya modelirovaniya dannykh naturnogo eksperimenta [Software complex for modeling data of a full-scale experiment]. Perspektivy nauki [Prospects of Science], 7 (106), pp. 39-43.
5. Overwiew. Available at
6. Prokhorenok N.A. (2012) Python 3 i PyQT5. Razrabotka prilozhenii [Python 3 and PyQT5. Application Development]. Saint Petersburg: BKhV-Peterburg Publ.
7. Psycopg – PostgreSQL database adapter for Python. Available at: https://www.psycopg.org/docs [Accessed 21/03/2022]..
8. QAbstractItemModel. Available at: https://pyside.github.io/docs/pyside/PySide/QtCore/QAbstractItemModel.html [Accessed 13/03/2022].
9. QAction. Available at: https://pyside.github.io/docs/pyside/PySide/QtGui/QAction.html [Accessed 18/03/2022].
10. QMenu. Available at: https://pyside.github.io/docs/pyside/PySide/QtGui/QMenu.html [Accessed 11/03/2022].
11. QTreeView. Available at: https://pyside.github.io/docs/pyside/PySide/QtGui/QTreeView.html [Accessed 13/03/2022].
12. Shenig Gans-Yurgen PostgreSQL 11 (2018). Masterstvo razrabotki.
13. SQLAlchemy. Available at: https://docs.python.org/3.9/tutorial [Accessed 13/03/2022].