

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	2
1 Методики прогнозирования состояния береговой зоны	9
1.1 Геоинформационные системы	12
1.2 Технологии ГИС	16
2 Проектирование ГИС для прогнозирования береговой зоны	20
2.1 Концептуальная модель предметной области	21
2.1.1 Этапы моделирования с использованием DSAS	30
2.2 Проектирование информационно-вычислительной инфраструктуры	32
2.2.1 Функциональные требования	33
2.2.2 Вычислительные процессы в ИВИ	33
2.2.3 Архитектура системы	34
2.2.4 Распознавание контура береговой линии	35
2.2.4.1 Библиотека Segment Anything	36
2.2.5 Анализ структуры объектов изображения	36
3 Реализация подсистем распределенной вычислительной среды	41
3.1 Модуль управления процессом моделирования	41
3.2 Интерфейс REST загрузки и обработки изображений	41
3.3 Исполнение задач с долгим временем вычисления	48
3.4 Хранение изображений	48
3.4.1 Стандарт HDF5: формат и инструментарий	48
3.4.2 Хранение изображений и масок	49
3.5 Тестирование	50
ЗАКЛЮЧЕНИЕ	55
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ	57
ПРИЛОЖЕНИЕ А Граф знаний, полученный анализом масок изображения, формат turtle	60

ВВЕДЕНИЕ

Фундаментальные исследования уникальных экологических систем, какой является береговая зона крупного водного объекта, проводимые в мире и России, базируются на мониторинге, хранении и обработке больших объемов научных пространственно-временных данных и знаний, а также на использовании распределенных информационно-вычислительных технологий и их приложений, современных сетей передачи данных.

Береговая зона крупного водохранилища. Водохранилища – регулируемые водохозяйственные и природные объекты, формирующие единую природно-техническую систему.

Объектом исследования является Братское водохранилище (рисунок 1). На карте красным цветом выделены участки береговой зоны с высокой интенсивностью абразионного размыва (ширина абразивного размыва 50-200 м).

При долгосрочном прогнозе абразионной переработки берегов Братского водохранилища на 25-летний период эксплуатации, сделанного Г. И. Овчинниковым в 90-х годах прошлого века [10], максимальные размывы (до 100 м) прогнозировались для участков, сложенных рыхлыми отложениями основной акватории правого берега водоема. Протяженность берегов Братского водохранилища составляет 6030 км, около 38 % сложено рыхлыми четвертичными отложениями. Абразионные берега составляют 2056 км (34.2 % общей протяженности берегов) [11]. Максимальная ширина размыва характерна для склонов, сложенных рыхлыми образованиями (пески, лессовидные супеси и суглинки), минимальные – для скальных и полускальных грунтов (песчаники, доломиты, известняки, алевролиты, аргиллиты, мергели) (рисунок 1).

Результаты долговременного мониторинга показали, что после более 50-ти лет эксплуатации Ангарских водохранилищ береговая зона все еще не достигла стадии устойчивого равновесия. Сохраняется стабильная динамика абразионного размыва, особенно береговых склонов, сложенных рыхлыми отложениями (рисунок 1).

На рисунке 2 показаны многолетние изменения уровня воды в водохранилище — основного фактора, вызывающего деградационные геологические процессы, частично находящегося под управлением технологических процессов каскада электростанций Иркутской области.

От стабильности береговой зоны зависит возможность ее технического, рекреационного и др. видов использования, особенно в условиях, когда уровень воды регулируется технически в достаточно большом диапазоне значений сезонного (2-3 м) и многолетнего регулирования (до 10 м).

За положение береговой линии принята последовательная линейная характеристика - положение бровки берегового уступа, которое изменяется под воздействием волновой деятельности водохранилища и процессов разрушения берегового уступа.

Для формирования политики использования береговой зоны необходимо проводить мониторинг геологических процессов береговой зоны водохранилищ, для этого необходимо проводить каталогизацию и хранение данных, включая исторические данные и современные. До использования спутниковых изображений для получения первичных данных о контуре береговой линии использовались полевые измерения и исследования и аэрофотосъемка. В настоящее время стали

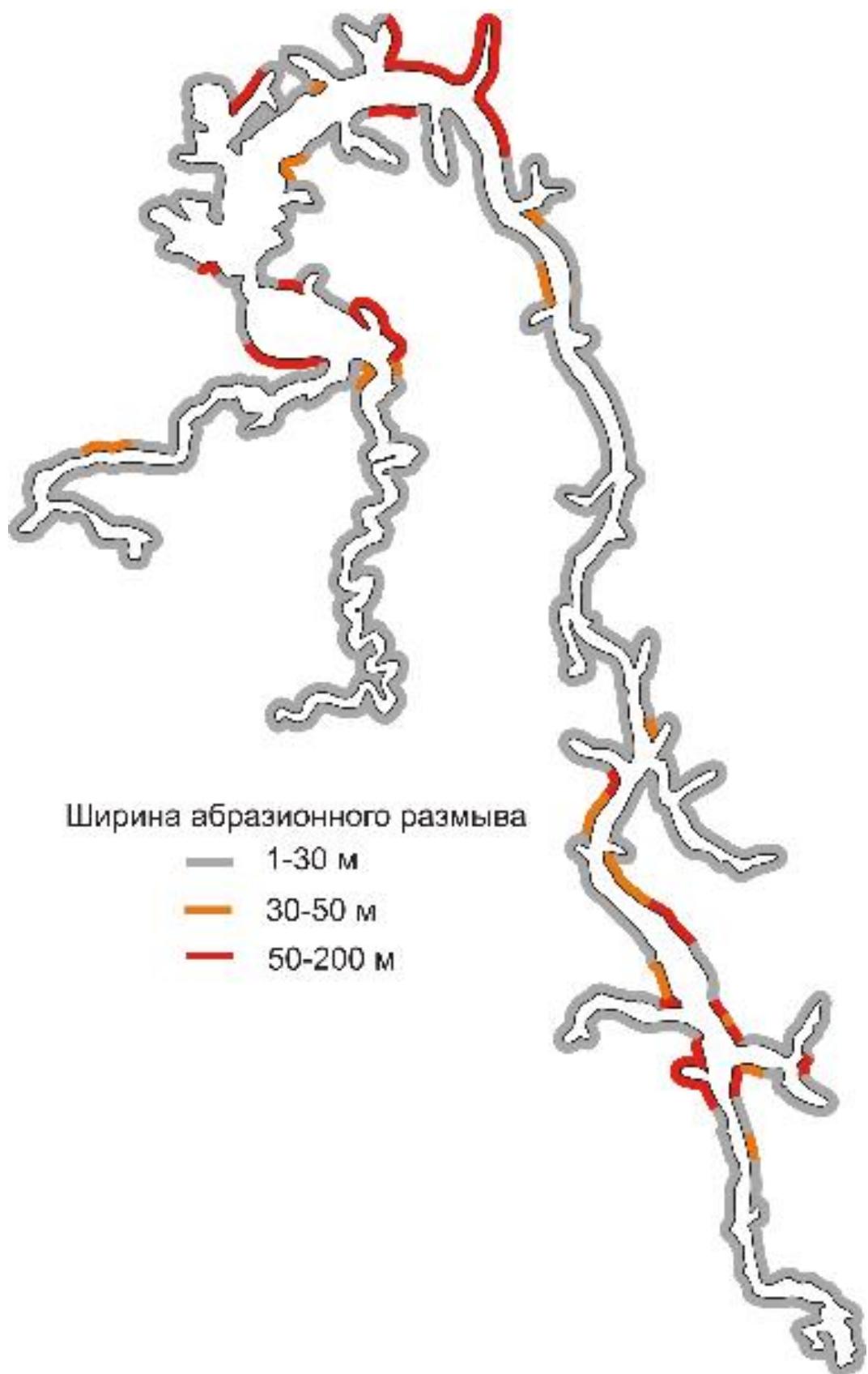


Рисунок 1 – Карта районирования берегов Братского водохранилища по ширине
абразионного размыва

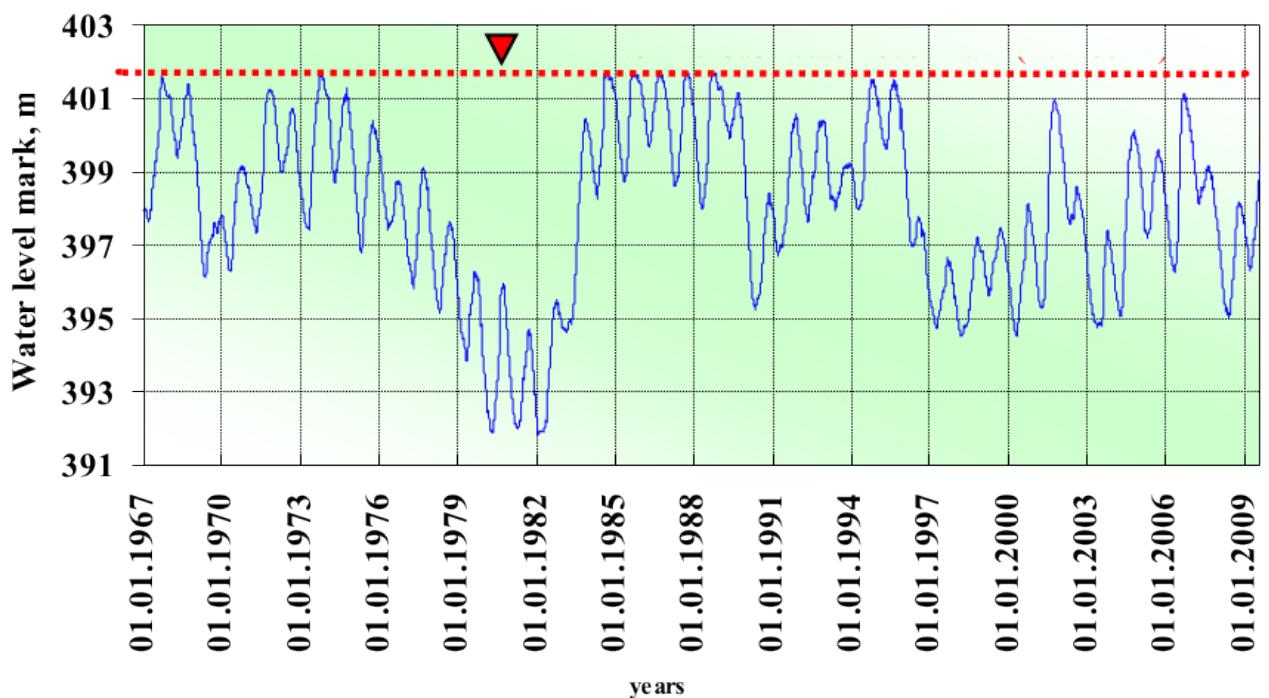


Рисунок 2 – Изменение уровня Братского водохранилища за период 1967-2009 гг (по данным метеостанции Балаганск)

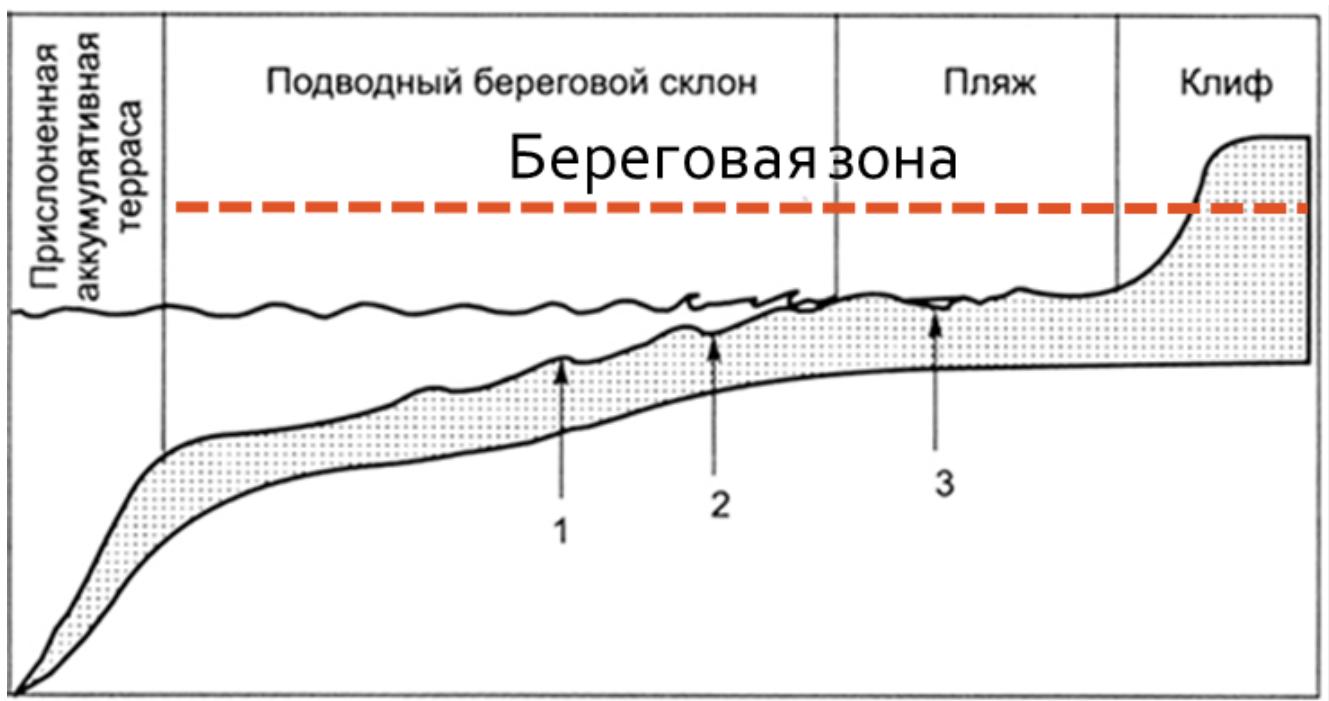


Рисунок 3 – Схема строения береговой зоны водоема

доступны съемка со спутника, получение ортофотопланов при помощи квадрокоптеров, измерения ключевых точек при помощи GPS/GLONASS.

Количество данных, таким образом, становится достаточно большим и требует цифрового представления в виде информационной системы, отражающей данные на некоторую координату. В связи с этим одним из важнейших компонентов информационных ресурсов являются геоинформационные системы, поскольку они позволяют согласованно представлять и системно анализировать информацию о географически связанных объектах природной среды и при этом собирать и хранить данные с автоматизированных средств наблюдений и классических средств сбора, осуществлять их автоматизированную обработку и представление в картографическом виде.

Для мониторинга краткосрочной и долгосрочной динамики изменений протяженных побережий в Европе [28], странах Юго-Восточной и Южной Азии [26],[?] применялась DSAS.

Цель работы - разработать геоинформационную систему крупного водного объекта для мониторинга береговой зоны.

Задачи

1. Проведение полевых исследований (снятие координат),
2. Оцифровка данных береговой линии (координат, спутниковых изображениях, аэрофотоснимков, ортофотопланов),
3. Реализация автоматизации некоторых этапов п.2 на основе современных технологий,
4. Организация доступа к внешним данным (уровни водохранилища),
5. Прогнозирование динамики береговой зоны,
6. Оценка влияния опасных факторов на технические и социальные объекты в береговой зоне.

Научная новизна разработанных технологий и полученных результатов заключается в использовании новых алгоритмов распознавания объектов на растровых изображениях (линии бровки берегового уступа) при условии ограниченности вычислительных ресурсов и информационного обеспечения.

Автоматизация распознавания береговой линии: обзор технологий

Фундаментальные исследования уникальных экологических систем, какой является береговая зона крупного водного объекта, проводимые в мире и России, базируются на мониторинге, основывающимся на больших данных. Мониторинг опасных экзогенных геологических процессов входит в единую информационную систему Государственного мониторинга состояния недр, который является составной частью государственного мониторинга состояния и загрязнения окружающей среды. Результаты долговременного мониторинга Братского водохранилища показали, что после более 50-ти лет эксплуатации Ангарских водохранилищ береговая зона все еще не достигла

стадии устойчивого равновесия. Разрушение берегов под воздействием природных геологических процессов (абразии, карста, эрозии, оползней) в значительной мере осложняет функционирование береговой зоны. От стабильности береговой зоны зависит возможность ее технического, рекреационного и др. видов использования, особенно в условиях, когда уровень воды регулируется технически в достаточно большом диапазоне значений сезонного (2-3 м) и многолетнего регулирования (до 10 м).

Наборы данных мониторинга характеризуются разнородностью и разноструктурированностью (электронные таблицы, тематические карты, космоснимки, 3D-модели, фото и видеоизображения), пространственно-временной привязкой, большим объемом и высокой скоростью роста объемов информации. Очевидно, что большая часть информации, связанной с мониторингом, является неструктурированной, т.е. для ее алгоритмической обработки необходимо решать сложные задачи распознавания ее структуры. Причем вычислительная сложность данного этапа предварительной обработки информации является очень высокой. В данном исследовании рассматривается проблема автоматизированного преобразования растрового представления изменяющегося во времени контура береговой линии равнинного водохранилища в векторный формат для дальнейшего мониторинга геологических процессов.

Аналогичные задачи в настоящее время являются актуальными, причем каждый конкретный случай характеризуется определенной спецификой, связанной с различными характеристиками входных изображений: качество снимков, масштаб объекта исследования, наличие предварительно обработанных вручную вариантов задач, мощность и сложность вычислительной инфраструктуры. Приведем несколько примеров, найденных в научных статьях.

В исследовании [2] предложен подход к поддержке решения задачи выявления и мониторинга береговых линий и оценке эрозии берегов. При помощи спутниковых снимков высокого разрешения территории Вьетнама, использовались модели глубокого обучения. Цели исследования: (1) предложить индикаторы для идентификации береговых линий и берегов; (2) построить модели глубокого обучения (DL) для автоматического дешифрирования береговых линий и берегов по снимкам дистанционного зондирования высокого разрешения; и (3) применить модели, обученные DL, для мониторинга эрозии берегов Вьетнама. Получены восемь DL-моделей на основе четырех вариантов искусственных нейронных сетей, включая U-Net, U2-Net, U-Net3+ и DexiNed. В качестве исходных данных для обучения моделей использовались изображения высокого разрешения, полученные с помощью программы Google Earth Pro. В результате U-сеть, использующая размер входного изображения 512×512 , обеспечила наивысшую производительность 98 % при функции потерь – 0.16. Результаты интерпретации этой модели использованы для идентификации береговой линии при оценке эрозии берегов из-за повышения уровня моря и штормов в течение 20 лет. Результаты показали, что линия уреза воды идеально подходит для наблюдения за сезонными приливными изменениями или непосредственными движениями волн, то линия бровки подходит для оценки береговой эрозии, вызванной влиянием повышения уровня моря и штормов. Данная работа сформировала отношение к тому, как модель U-Net может быть использована для прогнозирования изменений береговой линий государств, граничащих с мировым океаном.

Разработка автоматизированной масштабируемой методики выделения береговой линии

на основе спутниковых снимков в настоящее время ограничена низкой доступностью открытых, глобально распределенных и разнообразных размеченных данных, на основе которых можно разрабатывать и тестировать методики. В исследовании [3] представлен Sentinel-2 Water Edges Dataset (SWED), новый и специально разработанный набор маркированных изображений для разработки и тестирования методов автоматического извлечения данных о морфологии береговой линии, используя снимки Sentinel-2. SWED состоит из 16-ти размеченных сцен Sentinel-2 для обучения и 98-ми пар «метка-изображение» для тестирования, данные получены из разных мест мира и содержат примеры большого количества различных типов береговой линии, природных и антропогенных особенностей береговой линии. Для получения оценки качества распознавания для SWED обучены и протестированы четыре модели конволюционных нейронных сетей, основанных на архитектуре U-Net. Модели оптимизированы с использованием категориальной потери перекрестной энтропии, потери Сёренсена-Дайса и двух новых функций потерь, которые используются для фокусировки внимания обучения модели на границе вода-суша. С помощью гибридного процесса количественной и качественной оценки модели показано, что модель, обученная с использованием новой функции потерь Sobel-edge, обладает большей чувствительностью к мелкомасштабным, узким особенностям береговой линии, в то время как количественная производительность, продемонстрированная категориальной перекрестной энтропией, близка к максимальной. Набор данных SWED опубликован в открытом доступе для использования сообществами специалистов по дистанционному зондированию и машинному обучению, а потеря Sobel-edge доступна для использования в приложениях машинного обучения, где важна чувствительность к граничным особенностям.

Статья [4] рассматривает задачу сегментации морской территории (СМТ) – важной задачи дистанционного зондирования для различных береговых и экологических исследований, таких как выделение береговой линии, береговая эрозия, мониторинг прибрежных районов, обнаружение судов или айсбергов. Данное исследование направлено на улучшение производительности СМТ путем модификации модели Standard U-Net (SUN) и разработки системы автоматического выделения береговой линии. Модель SUN в целом имеет приемлемые характеристики для многих приложений, но вопросы повышения надежности выделения береговой линии остаются актуальными. В предложенной системе в первую очередь анализируются три различных входных изображения: красно-зелено-синие (RGB), нормализованный индекс разности воды (NDWI) и изображения ближнего инфракрасного диапазона (NIR). Затем для улучшения результатов сегментации в архитектуру SUN внесены изменения. Основные модификации заключаются в использовании различных функций потерь и двух методов слияния для RGB- и NIR-изображений. Результаты сегментации переданы в последующий конвейер автоматического выделения береговой линии на основе морфологических операций и анализа связности пикселей. Этапы обучения и тестирования выполнены на основе эталонного набора данных о прибрежных районах Китая. Кроме того, для оценки эффективности алгоритмов собран дополнительный набор данных, состоящий из временного ряда снимков Landsat-8 Южного побережья Каспийского моря. Результаты показывают, что предложенные изменения эффективно повышают производительность SUN, при этом наиболее значительное улучшение показателя «пересечения над объединением» (IoU), значение которого достигает 1,68% и 8,95% в наборах данных Китая и Каспийского моря, соответственно, а также

превосходит другие современные модели, включая FC-DenseNet и DeepLabV3+.

Таким образом, современные средства [2]-[4] автоматизации построения контуров береговых линий строятся с использованием нейронных сетей (НС) глубокого обучения, получаемых, в некоторых случаях, при помощи дообучения. Процесс обучения организуется на основе большого количества размеченных спутниковых изображений, разметка делается вручную, что достаточно трудоемко. Применяемый в данной ВКР подход основывается на использовании предобученной НС Segment Anything (SA), распознающей предметы (порядка миллиона классов) на изображении. SA не обучалась на распознавание береговых линий, но тестовые испытания показали хорошие результаты выделения контуров областей, граничащих с береговой линией. Чтобы определить контур, необходимо распознать объекты, включающие точки контура. Для этого необходимо провести процесс распознавания, с использованием дополнительной эвристической информатизации.

Процесс распознавания контура реализуется пошагово: 1) в ГИС (геоинформационной системе) выбирается интересующая область и контур береговой линии с векторной карты OpenStreetMap, данный контур является начальным приближением результата; 2) для выбранной области с серверов дистанционного зондирования загружаются изображения с разметкой по конкретным датам, изображения помещаются в серверное хранилище; 3) запуск системы SegmentAnything в режиме распознавания всех возможных объектов, результат распознавания в виде набора бинарных масок сохраняется на сервере; 4) алгоритмический анализ свойств объектов, представленных масками, и их расположение относительно друг друга и краёв изображения, результаты сохраняются в Абоксе онтологии на сервере; 5) анализ взаимного расположения объектов, выявление множества точек, относящихся к береговой линии; 6) построение контура по выявленным точкам (перевод в векторный формат), сохранение результата в shape-файл.

Преимущества предложенного подхода – это а) отсутствие трудоемкого этапа подготовки данных для обучения НС, б) распознавание контура представляется (программируется) в виде правил, что дает возможность управлять процессом распознавания, в частности «сцеплять» объекты или контуры с разных изображений; в) SA, ввиду обученности на большом количестве изображений не привязана к свойствам конкретных изображений (размеру, цветовой гамме, повороту и т.д.), что позволяет г) реализовывать (в перспективе) процедуры последовательного уточнения характеристик береговой линии, переходя к изображениям более высокого разрешения.

1 Методики прогнозирования состояния береговой зоны

Прогнозирование изменений геологической среды является одним из приоритетных научных и прикладных направлений в науках о Земле. Мониторинговые исследования за изменениями природно-технических систем и их отдельных элементов являются надежной базой для прогнозных оценок.

В 1972 году на Стокгольмской конференции ООН Р. Манном впервые было введено понятие глобального мониторинга окружающей среды (global environmental monitoring system — GEMS), которым он обозначил систему повторных наблюдений окружающей среды в пространстве и времени. С этого момента термин «мониторинг» вошел в нашу научную литературу и стал применяться в разных областях знаний, в том числе и в области геологии [24]. В области инженерной геологии стационарные наблюдения сопутствовали инженерным изысканиям на всех сложных объектах. Первая оползневая станция ЦНИГРИ в Крыму была организована в 1930 году.

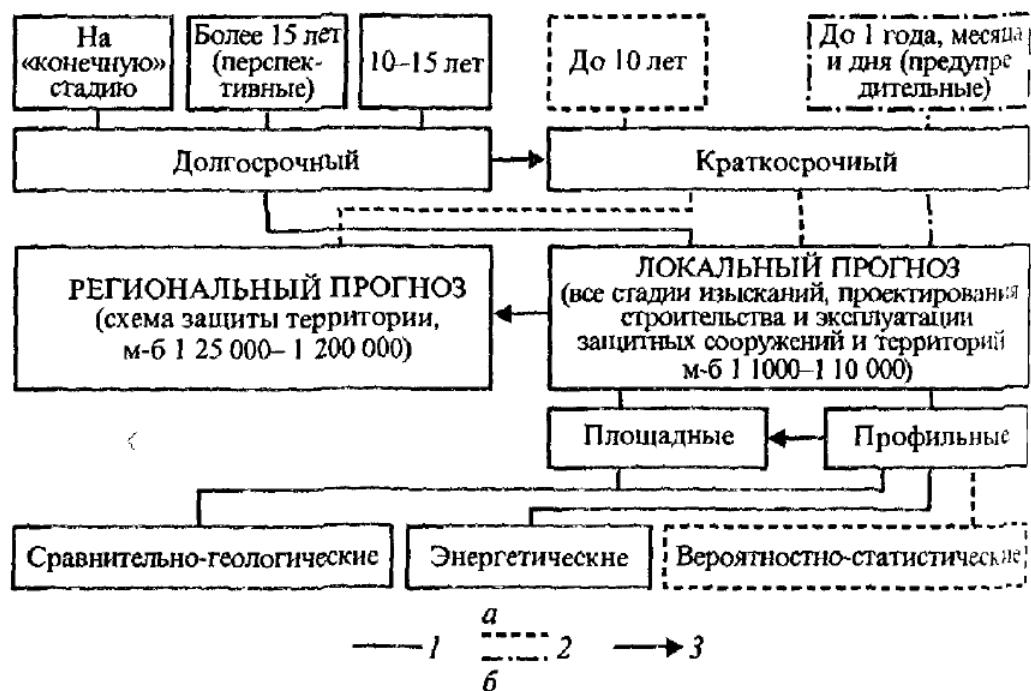
Для Ангарского каскада ГЭС лабораторией инженерной геологии и геоэкологии ИЗК СО РАН была создана система геодинамических полигонов и стационаров, расположенных в различных ландшафтных условиях. Систематические ежегодные наблюдения за отдельными видами ЭГП проводились на 77 участках, среди которых на 43 изучалась абразия, на 15 — гравитационные процессы, на 11 — карстово-суффозионные и на 8 — эрозионные [25].

Целью мониторинговых исследований является изучение режима и условий функционирования локальных береговых геосистем и их изменения под воздействием техногенных факторов, связанных с созданием и режимом эксплуатации водохранилища.

В нашей стране наиболее применяемыми в настоящее время являются методы, предложенные Е. Г. Качугиным (1959), Г. С. Золотаревым (1969), Н. Е. Кондратьевым (1960), Е. К. Гречишевым (1961) Л. Б. Розовским, И. П. Зелинским (1975), И. А. Печеркиным (1969), Институтом земной коры СО РАН (1976 г.), ПНИИСом (Рагозин и Бурова, 1993) и другими научными коллективами. По различным критериям все виды прогнозов переработки берегов водохранилищ могут быть объединены в три разных класса (рисунок 1.1) [24]. Так, временные прогнозы подразделяются на краткосрочные (предупредительные) с периодами прогнозирования от 1 до 10 лет и долгосрочные (перспективные), включающие прогнозы на 10 и более лет, в том числе на конечную стадию развития берега. Пространственные прогнозы объединяют в себе локальные, характеризующие переработку по отдельным профилем (профильный прогноз) или участкам (площадной прогноз) и выполняемые в масштабах 1:1000—1; 10000; и региональные, характеризующие переработку по периметру всего водохранилища или его отдельной части и производимые в масштабах 1:25 000—1:200000. По теоретическим предпосылкам обоснования расчетов все методы подразделяются на три группы — энергетические, геологического подобия и вероятностно-статистические (или стохастические, по Г. С. Золотареву (1990)).

В научно-технической литературе известны три группы методов прогнозной оценки переработки берегов водохранилищ:

1. энергетические ([19], [20] и др.);



Прогнозы, выполняемые на водохранилищах: 1 — проектируемых и эксплуатируемых; 2 — эксплуатируемых: а — на всех стадиях изысканий и проектирования инженерной защиты, б — в процессе строительства и эксплуатации; 3 — последовательность выполнения прогнозов.

Рисунок 1.1 – Схема прогнозов процессов переработки берегов водохранилищ при инженерных изысканиях для освоения и защиты прибрежные территории [22]

2. сравнительно-геологические ([15], [16] и др.);
3. вероятностно-статистические [17].

Энергетические методы при прогнозе преимущественно учитывают энергию волнения (гидрологический фактор), область применения их ограничена простыми и однородными в инженерно-геологическом отношении условиями, характеризуются низкой точностью прогнозной оценки. Для их обоснования требуются значительные трудозатраты по каждому расчетному поперечнику. Применимы только на первой стадии переработки берегов действующих водохранилищ.

Сравнительно-геологические методы основаны на аналогиях природно-техногенных условий водохранилищ. Основные трудности, снижающие точность прогнозных оценок в этих методах связаны с подбором надежных аналогов и выбором критериев подобия, что редко удается сделать в реальных условиях.

Метод вероятностно-стохастических моделей [17] по своей технической сущности наиболее близок к предлагаемому техническому решению (прототип). Этот метод заключается в учете зависимостей переработки берегов от различных природно-техногенных факторов по данным режимных наблюдений с построением стохастических моделей процесса. Его основной недостаток, несмотря на большие трудозатраты снижающий точность прогнозной оценки, заключается в недочете общих закономерностей развития процесса, стадийности формирования берегов, требует большого объема режимных наблюдений по каждому расчетному поперечнику в отдельности.

В Институте земной коры СО РАН Е. К. Гречищевым (1961). был разработан энергетический метод, позволяющий производить прогноз ширины зоны размыва на разное количество лет. Для прогноза необходимы сведения по ветровому волнению, определяющему энергию, знание уровенного режима водоемов, точные инженерно-геологические разрезы по расчетным профилям и данные по размываемости грунтов. Ширина зоны размыва определяется через объем размытых горных пород.

Позже по результатам исследований динамики берегов водохранилищ Ангарского каскада ГЭС и выявленных закономерностей развития береговой зоны был сделан прогноз размыва берегов на 25-летнюю стадию развития [10]. Расчеты производились по методике, разработанной в Институте земной коры СО РАН, учитывающей весь спектр факторов (геолого-геоморфологические, морфометрические, гидродинамические и техногенные), влияющих на динамику берегов. Кроме того, использовались прогнозно-диагностические модели динамики берегов, построенные на основе многолетних рядов наблюдений по стационарным участкам, расположенным в различных геолого-геоморфологических и гидродинамических условиях. Учет влияния колебания уровня воды в водохранилищах оценивался его повторяемостью на определенных отметках в пределах выделенных полутораметровых подзон. Прогноз был выполнен с учетом того, что в 50 % случаев уровень воды будет находиться в пределах верхней подзоны размыва.

1.1 Геоинформационные системы

Геоинформационные системы (ГИС) - это автоматизированные системы, функциями которых являются сбор, хранение, интеграция, анализ и графическая интерпретация пространственно-временных данных, а также связанной с ними атрибутивной информации о представленных в ГИС объектах.

Классификация ГИС [31]. Географические информационные системы классифицируют по-разному в зависимости от масштабности и функционала, а также других признаков.

По территориальному охвату ГИС бывают:

- глобальными;
- субконтинентальными;
- национальными;
- региональными;
- субрегиональными;
- локальными или местными.

По уровню управления:

- федеральными;
- региональными;
- муниципальными;
- корпоративными.

По функциональности:

- полнофункциональными;
- для просмотра данных;
- для ввода и обработки данных;
- специализированными с дополнительными функциями.

По предметной области:

- картографическими;
- геологическими;
- городскими или муниципальными;

- природоохранными,
- туристическими.

Если в ГИС присутствуют возможности цифровой обработки изображений, то такие системы называются интегрированными ГИС (ИГИС). Полимасштабные, или масштабно-независимые ГИС обеспечивают графическое или картографическое воспроизведение данных в любом масштабе с наибольшим разрешением. Пространственно-временные ГИС работают с данными во времени.

Предметом исследования ВКР является процесс сбора данных и построения прогноза изменения береговой линии водохранилища.

Для обеспечения оценки состояния береговой зоны и ее перспективного состояния может быть оценено при помощи получения дополнительных данных в результате математического моделирования. Процесс размывания/накопления грунта зависит от комплекса природных и внешних факторов таких как:

- Положение и продолжительность стояния уровня водохранилища;
- Геолого-геоморфологические условия строения береговой зоны;
- Ветро-волновая нагрузка на участке берега размыва.

Самым точным емким информационным подходом является моделирование процесса размыва как физического процесса. Такой подход требует большое количество данных и построение сложных алгоритмов моделирования, а также много вычислительных ресурсов. В результате получается краткосрочный прогноз.

В данной работе использована модель грубой оценки контура береговой линии на основе проведения экстраполяции при помощи метода, реализованного в библиотеке DSAS[26], реализованной в виде модуля ArcGIS. Исходный код является открытым и представлен на языке C#.

Digital Shoreline Analysis System (DSAS) DSAS v5.0 - это модуль для ArcGIS 10, разработанный совместно Геологической службой США (USGS) и TPMC Environmental Services [?], [?]. Разработанный алгоритм данного программного обеспечения позволяет пользователю вычислить скорость изменения береговой линии, используя географически привязанные береговые линии временных рядов в каждом разрезе с заданным пользователем временным интервалом.

DSAS использует пять операций, в частности, огибающую изменения береговой линии (shoreline change envelope SCE), чистое движение береговой линии (net shoreline movement NSM), скорость конечной точки (end point rate EPR), скорость линейной регрессии (linear regression rate LRR) и наименьшую медиану квадратов (least median of squares LMS) для оценки скорости изменения береговой линии в долгосрочной и краткосрочной перспективе.

За положение береговой линии принимается любая, указанная пользователем, линейная характеристика, такая как линия растительности, линия высокой воды, линия низкой воды или линия влажности/сухости. В данном случае за положение береговой линии принимается положение бровки берегового уступа, которое изменяется под воздействием волновой деятельности водохранилища и процессов разрушения берегового уступа.

- Прогноз DSAS смотрится при помощи экстраполяции точек контура, исходя из данных о контурах линий, соответствующих определенным датам съемки. Подход обладает рядом преимуществ:
- Не требует больших вычислительных ресурсов,
- Не требует данных о конкретных геологических процессах,
- Прост в реализации,
- Для модели не требуется большое количество данных,
- Данные могут охватывать большие диапазоны дат,
- Реализует прогноз на далекую перспективу,
- Случайные техногенные воздействия “усредняются” повторяющимися постоянными природными факторами.

К недостаткам модели относятся следующие:

- Модель дает достаточно грубую оценку расположения контура береговой линии,
- Достаточно сложно использовать при наличии периодически меняющихся факторов воздействия (см. например, проблематику моделирования уровня Каспийского озера).

Факторы воздействия на береговую линию Братского водохранилища характеризуются комплексностью, одним из которых является антропогенность, а именно, уровень воды.

Береговые линии могут быть оцифрованы из различных источников (например, цифровые ортофотопланы, привязанные исторические карты береговой съемки или спутниковые снимки), собраны с помощью полевых исследований с глобальной системой позиционирования или извлечены из лидарных исследований. Приведем примеры исходных данных. На рисунке ?? представлены два варианта: а) точки контура бровки, отснятые в поле (2008 год) и б) изображение берега, полученное с серверов Google Maps.

Более точными исходными данными являются спутниковые снимки высокого разрешения и отснятые и обработанные изображения, полученные при помощи квадрокоптера (ортографопланы). Сервера спутниковых снимков высокого разрешения позволяют также получать информацию не только для текущего года, но и за предыдущие годы, однако их достаточно трудно получить ввиду их частичной открытости. Ортофотопланы (рисунок ??) можно снимать каждый год, они дают приемлемое разрешение, а также для осуществления съемки не требуется больших материальных затрат.

Каждый вектор береговой линии представляет собой определенное положение во времени и имеет дату в таблице атрибутов класса характеристик береговой линии (таблица 1.1). Измерительные трансекты, откладываемые DSAS от базовой линии, пересекают векторы береговой



Рисунок 1.2 – Контур береговой линии представлен координатами точек и изображением с сервера

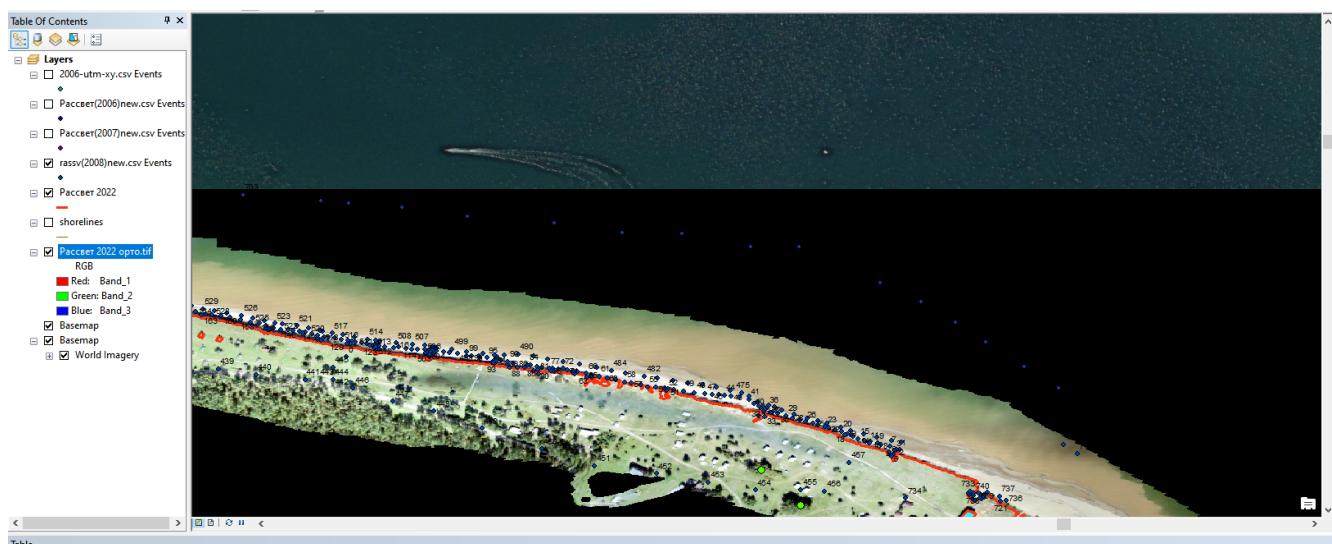


Рисунок 1.3 – Контур береговой линии представлен координатами точек и изображением с сервера

Таблица 1.1 – Атрибуты шейп-файла shorelines для Digital Shoreline Analysis System (DSAS) version 5.0

Field name	Data type	Attribute addition
OBJECTID	Object ID	Autogenerated
SHAPE	Geometry	Autogenerated
SHAPE Length	Double	Autogenerated
DATE	Text (Length=10 OR Length=20)	User-created
UNCERTAINTY		User-created
SHORELINE_TYPE		User-created

линии (рисунок ??). Точки пересечения предоставляют информацию о местоположении и времени, используемую для расчета скорости изменения. Расстояния от базовой линии до каждой точки пересечения вдоль трансекты используются для расчета выбранных статистических данных.

В связи с вышеперечисленным применение именно этой модели позволит сделать первый шаг в построении системы поддержки принятия решения о характере деградационных процессов береговой зоны, и, таким образом, поможет представить необходимую информацию муниципальному хозяйству данные о политике использования того или иного берега.

Как было сказано ранее одной из проблем использования математического моделирования - это информационное обеспечение, т.е. создание возможности подачи исходных данных в процесс моделирования. Для применения DSAS необходимо представить набор файлов требуемой структуры - Shapefile с контурами береговых линий, и соответствующий ему файл данных - значения атрибутов. Контуры и атрибуты могут быть получены при помощи привязки и оцифровки контуров в авиа- и спутниковых снимков. Процедура оцифровки достаточно трудоёмкая, и в данной ВКР предлагается подход к ее частичной автоматизации на основе анализа результатов приложения современных методов распознавания объектов на изображении.

1.2 Технологии ГИС

Преимущество ГИС в том, что эта система позволяет преодолеть основные недостатки обычных карт - их статичность и ограниченную емкость как носителя информации. В последние десятилетия бумажные карты из-за перегруженности информацией становятся нечитабельными. ГИС же обеспечивает управление визуализацией информации. ГИС в отличие от других информационно-аналитических систем предназначена для обработки и анализа пространственных данных. Информация об этих пространственных данных в цифровой форме называется геоинформацией.

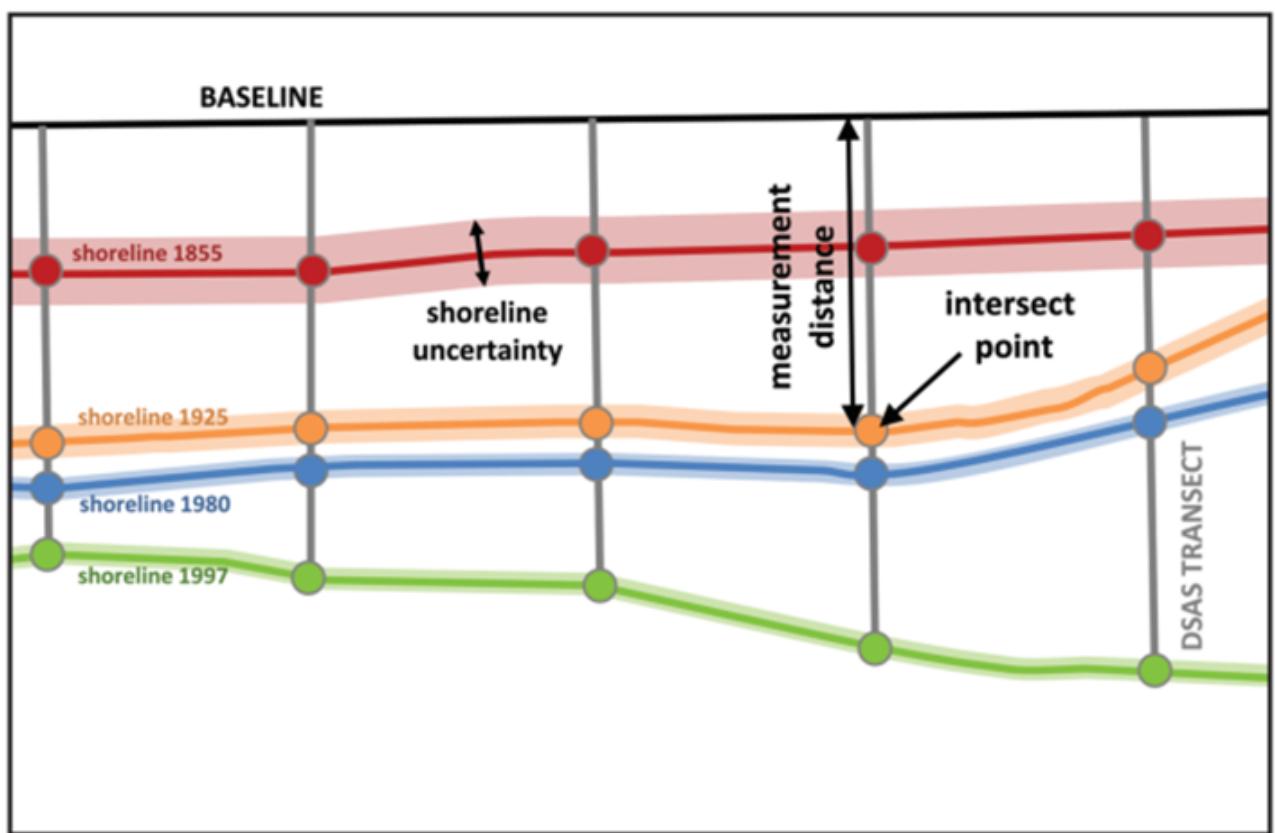


Рисунок 1.4 – Трансекты DSAS

мацией.

Тематически специализированная прикладная ГИС включает в себя пять ключевых составляющих: аппаратные средства, программное обеспечение, данные, исполнители и методы. ГИС-приложения объединяют различные типы информации, среди которых: картографические данные — представлены в виде карты и могут включать такую информацию, как расположение рек, дорог, жилых и нежилых строений; аэрофотоснимки и обычные фотографии и данные со спутников; данные дистанционного зондирования (обычно с применением воздушных шаров и дронов); глобальные системы позиционирования (GPS); данные из Интернета; документы, включая архивные таблицы и каталоги координат; данные из других



Рисунок 1.5 – Процесс создания ГИС

Технология ГИС позволяет накладывать все типы информации, независимо от их источника или исходного формата, поверх друг друга на одной карте. ГИС использует местоположение в качестве ключевой переменной, чтобы связать эти, казалось бы, несвязанные данные. Ввод информации в ГИС называется сбором данных. Информацию, которая уже находится в цифровой форме, можно просто загрузить в систему. Однако сначала карту необходимо отсканировать или преобразовать в цифровой формат (оцифровка данных).

Географические информационные системы включают три компонента:

Данные: ГИС хранит данные о местоположении в виде слоев информации по разным темам. Каждый набор данных имеет таблицу атрибутов, в которой хранится информация об объекте. Два

основных типа формата файлов ГИС — растровый и векторный. Растровый представляет собой сетки из ячеек или пикселей. Он полезен для хранения различных ГИС-данных. Векторный формат выглядит как многоугольник, в котором используются точки (называемые узлами) и линии. Векторные файлы нужны для хранения данных ГИС с четкими границами, такими как городские округа или улицы. В итоге технология позволяет отображать пространственные и линейные зависимости. Пространственные показывают топографию местности (поля, ручьи), а линейные представлены дорогами или коммунальными сетями.

Аппаратный компонент, который запускает программное обеспечение ГИС. Это может быть что угодно: мощные серверы, мобильные телефоны или персональные рабочие станции. Как правило, в работе с ГИС нужны два монитора, дополнительное хранилище данных и графические карты высокой четкости.

Программное обеспечение. Оно специализируется на пространственном анализе с использованием математики в картах. Такое ПО сочетает в себе географию с современными технологиями для измерения, количественной оценки и анализа. В ГИС. обычно используют языки программирования Python, SQL, C ++, Visual Basic и JavaScript.

В ГИС информация со всех различных карт и источников должна соответствовать одному масштабу — соотношению между расстоянием на карте и фактическим расстоянием на Земле. При этом разные карты имеют разные проекции. Чтобы перенести изогнутую трехмерную форму на плоскую поверхность, неизбежно требуется растяжение одних частей и сжатие других. Так, на карте мира могут быть показаны либо страны правильного размера, либо их правильные формы, но нельзя отобразить эту информацию одновременно. ГИС берет данные с разных карт мира и объединяет ее, чтобы отобразить в одной общей проекции. Одними из популярных программ ГИС считаются ArcGIS и QGIS.

2 Проектирование ГИС для прогнозирования береговой зоны

Анализ предметной области, представленный в Главе 1, показал, что для получения качественно новых результатов научных исследований в инженерной геологии береговых зон внутренних водоемов, необходимо создать ресурсы хранения, преобразования, обеспечения эффективного доступа к этой информации, а также ее визуализации. Перечисленным свойствам удовлетворяют программные системы, называемые *информационными системами* (ИС). Наличие пространственной привязки данных, т.е. ассоциирование конкретной информации с точкой или каким-то объектом на поверхности Земли, переводит ИС в класс *географических информационных систем*. Если элементы (*функциональные блоки*) ИС исполняются на отдельных вычислительных узлах или узлах хранения данных локальной вычислительной сети, и даже в отдельных процессах одного вычислительного устройства, то какая ИС относится к классу *распределенных информационных систем* (*распределенных программных комплексов*), а совокупность сетевых ресурсов, вовлеченных в информационно-вычислительные процессы называется *информационно-вычислительной инфраструктурой* (ИВИ).

В настоящее время элементы ИВИ представляют в виде сервисов, т.е. функциональных блоков, выполняющих заданные операции, получая исходные данные из сети (с другого сервиса) по определенному протоколу, и отправляющие результат также в сеть (на другой сервис). Экстремальным вариантом сервисной ИВИ является так называемая *микросервисная архитектура*. Набор реализуемых функций в таких сервисах минимизирован, и распределенный программный комплекс реализуется композицией большого числа таких микросервисов, сконфигурированных согласно дизайну ИВИ. Преимуществом такого подхода является больший потенциал к повторному использованию кода, поддержка стандартизации между отдельными проектами, более гибкой интеграции с другими ИВИ. Кроме того, функционирование процессов микросервисов, как правило, более предсказуемо, что дает возможность более надежного и качественного управления ИВИ.

При проектировании ИВИ решается задача управления распределенным вычислительным процессом – *синхронизацией* работы *сервисов*. Стандартный способ синхронизации работы сервисов в ИВИ – это использование специальных сервисов, называемых *брокеров сообщений* (*message brokers*). Брокеры сообщений позволяют конфигурировать ИВИ и управлять этой средой таким образом, чтобы логика вычислительного процесса соответствовала реализуемому алгоритму обработки данных. Также этот сервис обеспечивает перезапуск процессов в ИВИ в случае перезапуска узлов и локальной вычислительной сети. Еще одна функция – это распределение и, в некоторых случаях, перераспределение, балансировка вычислительной нагрузки между узлами, реализующими один и тот же сервис.

Таким образом, представленные в Главе 1 проблемы решаются при помощи реализации распределенной вычислительной среды обеспечения сервисов хранения и обработки данных, где некоторыми функциональными блоками выступают геоинформационные системы.

Таким образом, для обеспечения возможности проведения научных исследований в обла-

сти инженерной геологии береговых зон водохранилищ необходимо создать ресурсы хранения информации и вычислительные ресурсы, обеспечивающие продуктивную среду прогнозирования с использованием различных математических моделей. Перспективные, в смысле реализации в такой среде, ранжируются по видам задач, масштабу исследуемого объекта, размеру интервала времени, степени точности.

2.1 Концептуальная модель предметной области

Основная цель исследований, в рамках которых получены результаты данной ВКР, заключается в создании системы поддержки принятия решения (СППР) об оценке состояния береговой зоны водохранилищ, включая опасные геологические процессы. СППР, реализуемые с использованием функциональных блоков моделирования и автоматизированной оценки результатов, формирует базу создания *систем поддержки принятия решений* автоматизации выработки *антиинтуитивных решений*¹.

Процесс принятия решений опирается на концептуальное моделирование предметной области. В настоящее время стандартным методом представления концептуальных моделей являются онтологии, явные спецификации концептуального уровня представления предметной области [1].

Спроектированная онтология (рисунки??, 2.2), описывает предметную область инженерной геологии, связанную с развитием и мониторингом экзогенных процессов на берегах водохранилищ. Водохранилища являются искусственно созданными водоемами в отличие от озер. После их создания (в результате строительства ГЭС на реках) на берегах стали активно развиваться экзогенные процессы. На отдельных участках берега исследователи изучают формы экзогенных процессов (воронка, овраг, оползень, эоловая форма), геоморфологические и геологические условия их развития. Для каждого участка определены координаты (*latitude*, *longitude*, протяженность в метрах). В зависимости от преобладающих процессов можно определить генетический тип берега. Онтология содержит: 34 класса (рисунок 2.4), 24 свойств и 21 индивидуумов (рисунок 2.12). Свойства включают примитивные (data property) – 13 (рисунок 2.9) и объектные (object property) – 11 (рисунок 2.11).

К важным терминам предметной области относятся термины:

- природный объект;
- водоем;
- берег;
- участок;
- форма экзогенного процесса.

Их иерархия показана на рисунке 2.5.

¹ Антиинтуитивные решения принимаются на основе анализа имеющейся информации, а не на основе интуиции лица, принимающего решения

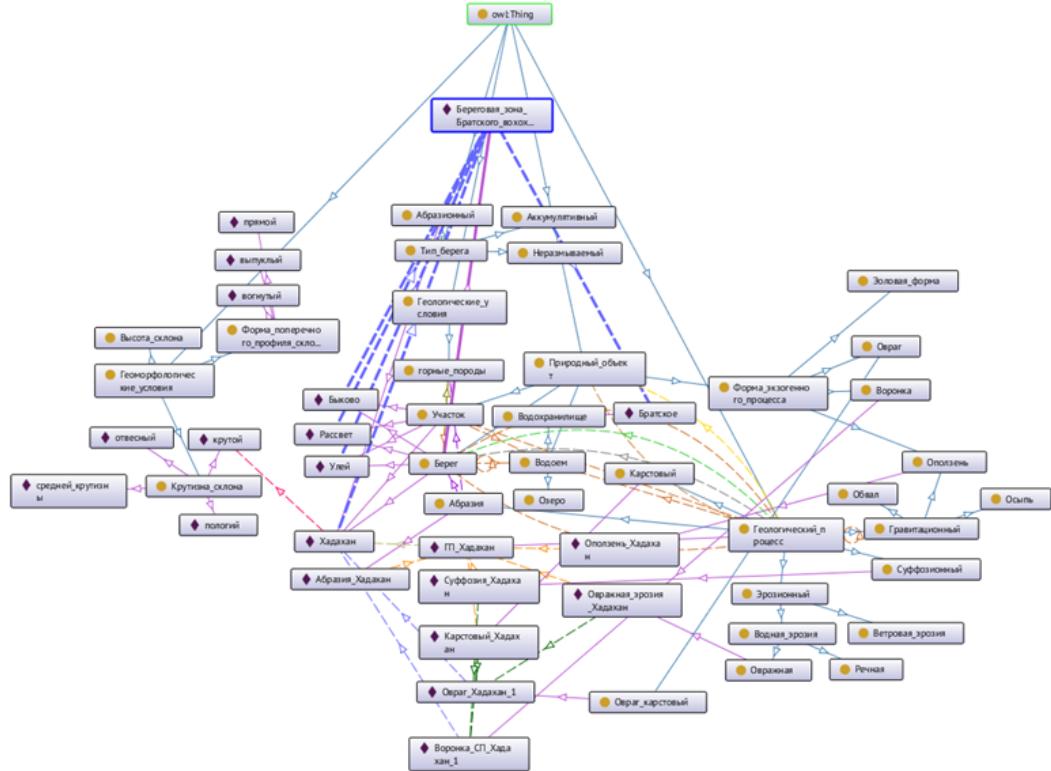


Рисунок 2.1 – Скриншот экрана с созданной онтологией в OntoGraph

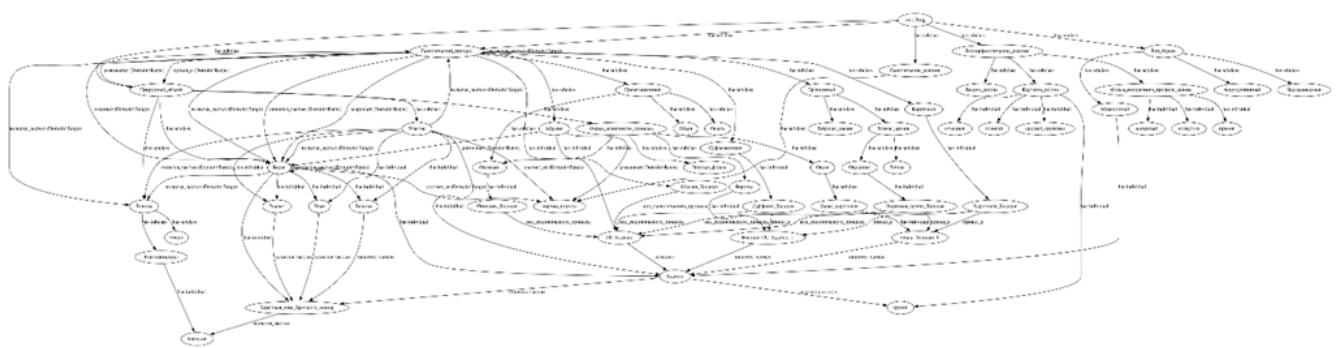


Рисунок 2.2 – Графическое изображение созданной онтологии, построенная в программе xdot

Ontology metrics:	
Metrics	222
Axiom	222
Logical axiom count	145
Declaration axioms count	77
Class count	34
Object property count	11
Data property count	13
Individual count	21
Annotation Property count	0
Class axioms	
SubClassOf	30
EquivalentClasses	0
DisjointClasses	1
GCI count	0
Hidden GCI Count	0
Object property axioms	
SubObjectPropertyOf	8
EquivalentObjectProperties	0
InverseObjectProperties	0
DisjointObjectProperties	0
FunctionalObjectProperty	0
InverseFunctionalObjectProperty	0
TransitiveObjectProperty	0
SymmetricObjectProperty	0
AsymmetricObjectProperty	0
ReflexiveObjectProperty	0
IreflexiveObjectProperty	0
ObjectPropertyDomain	10
ObjectPropertyRange	10
SubPropertyChainOf	0
Data property axioms	
SubDataPropertyOf	9
EquivalentDataProperties	0
DisjointDataProperties	0
FunctionalDataProperty	2
DataPropertyDomain	9
DataPropertyRange	10
Individual axioms	
ClassAssertion	26
ObjectPropertyAssertion	18
DataPropertyAssertion	12
NegativeObjectPropertyAssertion	0
NegativeDataPropertyAssertion	0
SameIndividual	0
DifferentIndividuals	0
Annotation axioms	
AnnotationAssertion	0
AnnotationPropertyDomain	0
AnnotationPropertyRangeOf	0

Рисунок 2.3 – Скриншот панели «Ontology metrics»

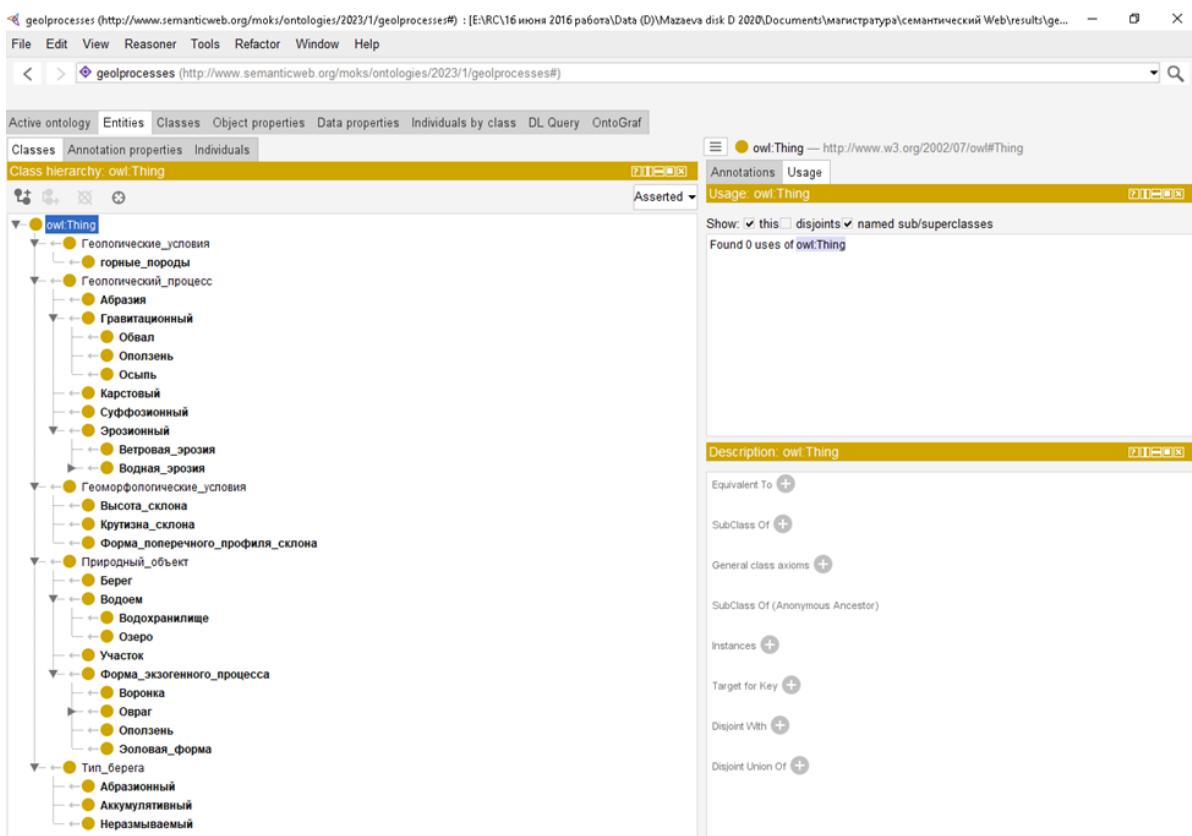


Рисунок 2.4 – Скриншот экрана с вкладками созданных классов

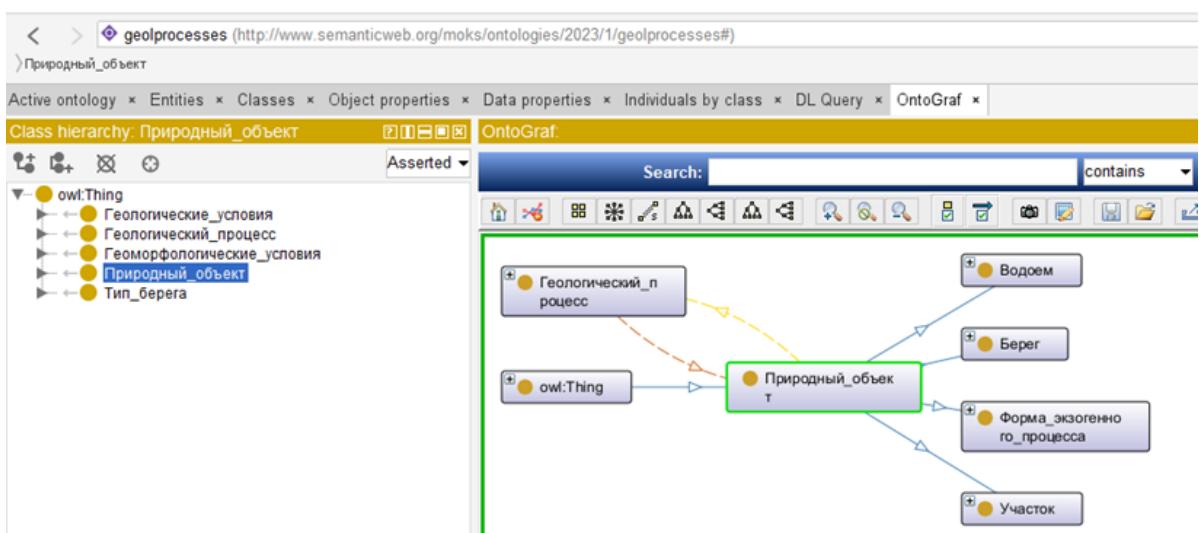


Рисунок 2.5 – Иерархия важных предметных терминов и ее отображение в OntoGraph

Далее представлены скриншоты экранных форм с вкладками созданных свойств и индивидуумов (рисунок 2.9).

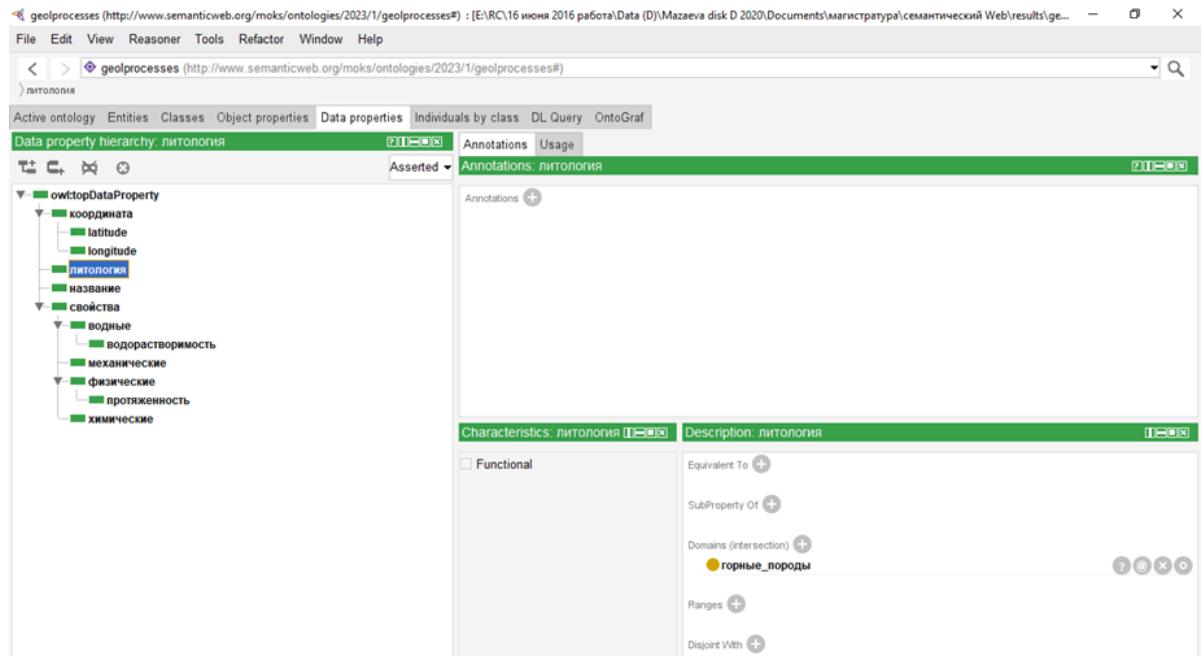


Рисунок 2.6 – Скриншот экрана панели «Data properties» с параметрами свойства для Литология

Из существующих онтологий в близкой предметной области была найдена онтология UMBEL [32], описывающая Natural Phenomena (рисунок 2.13). В этой онтологии они относятся к SuperType и им дано определение: NaturalPhenomena skos:definition «Этот SuperType включает в себя природные явления и естественные процессы, такие как погода, выветривание, эрозия, пожары, молнии, землетрясения, тектоника и т.д. Облака и погодные процессы включены особо. Также сюда входят климатические циклы, общие природные явления (например, ураганы), которые не имеют конкретных названий, и биохимические процессы и траектории.»

Созданная онтология «Экзогенные процессы на берегах водохранилищ» может быть расширена добавлением водохранилищ, участков, форм экзогенных процессов на участках. Углублена добавлением информации по геологическим и геоморфологическим условиям развития процессов. При добавлении GPS-привязки может использоваться как геоинформационная справочная система для заинтересованных лиц и организаций, а также как база данных для ведения мониторинговых (повторных) наблюдений за экзогенными процессами на берегах водохранилищ.

Данную разработку онтологии можно отнести к комбинированной. Ее создание началось с класса «Водоем» к которому относятся подклассы природных водоемов – «Озеро» и искусственно созданных – «Водохранилище». После были созданы классы «Берег» «Участок» «Форма экзогенного процесса». Все они были объединены в класс «Природный объект» (рисунок 2.4).

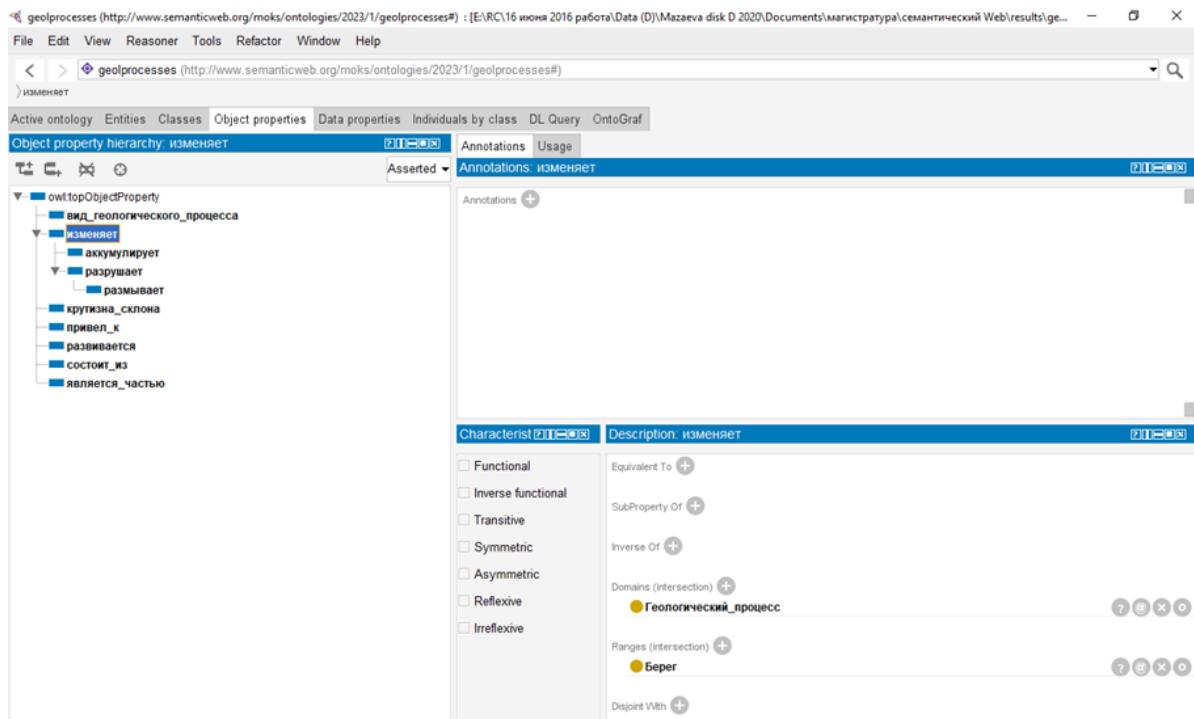


Рисунок 2.7 – Скриншот экрана панели «Object properties» с параметрами свойства для «изменяет»

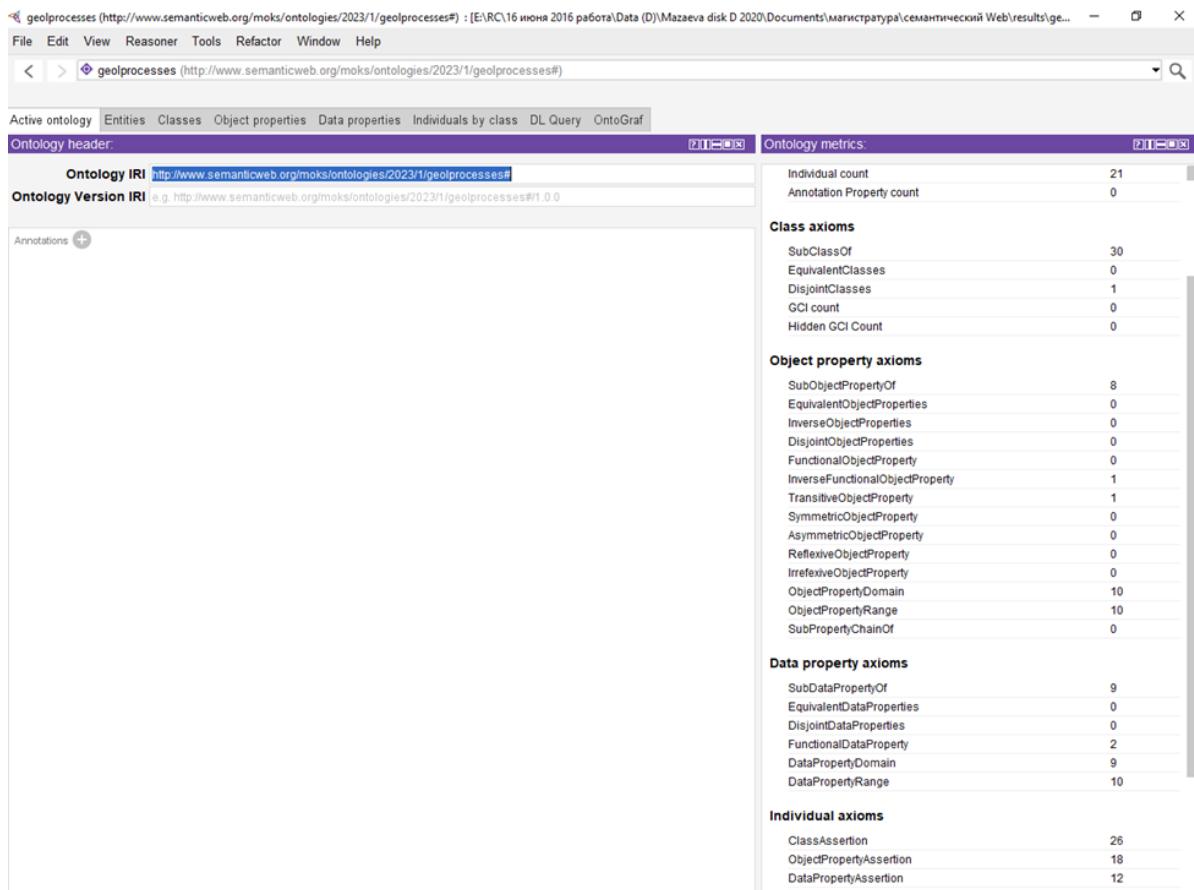


Рисунок 2.8 – Скриншот экрана панели Ontology metrics с указанием характеристик свойств

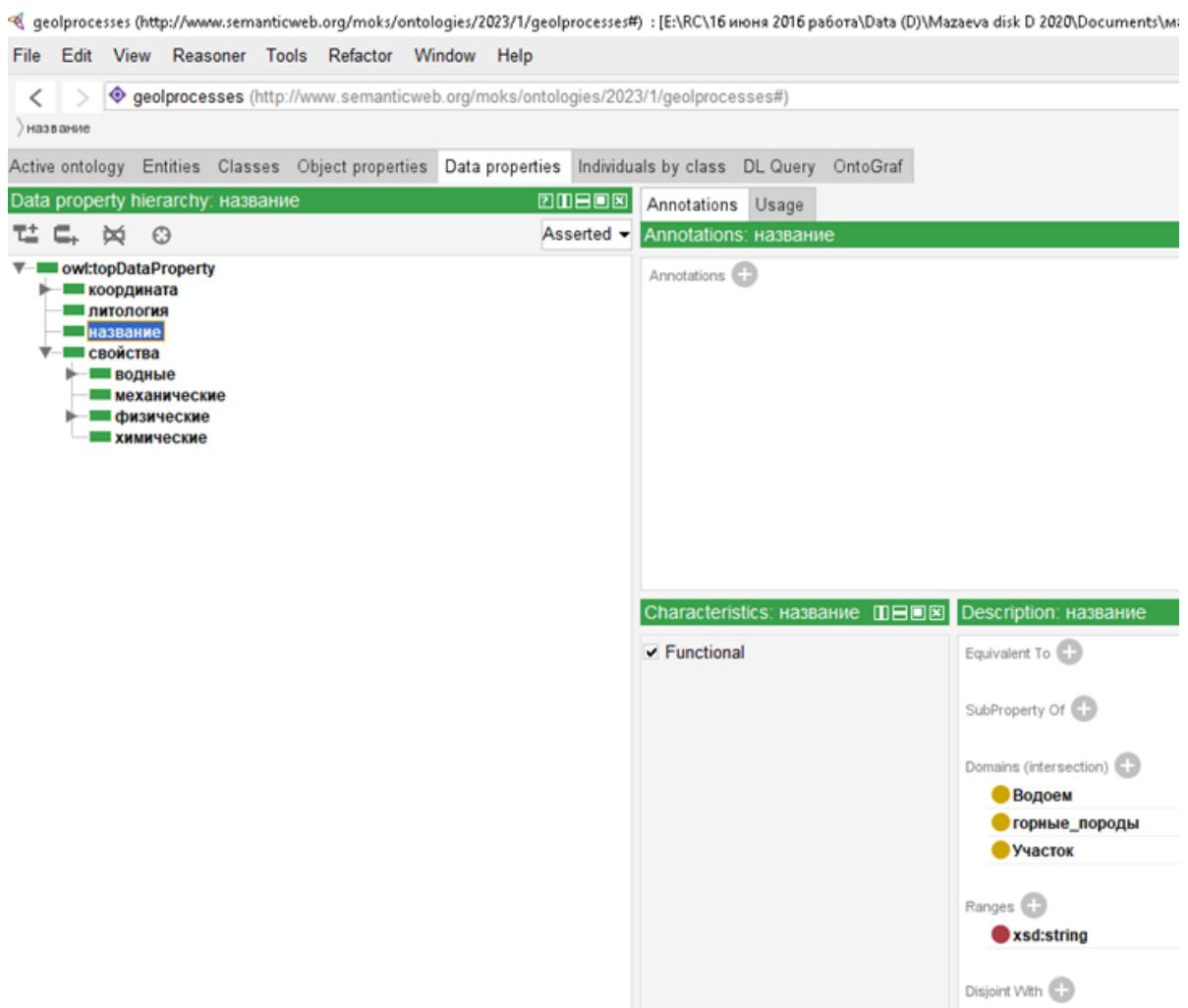


Рисунок 2.9 – Скриншот экрана панели “Data property” с примером функционального свойства «название»

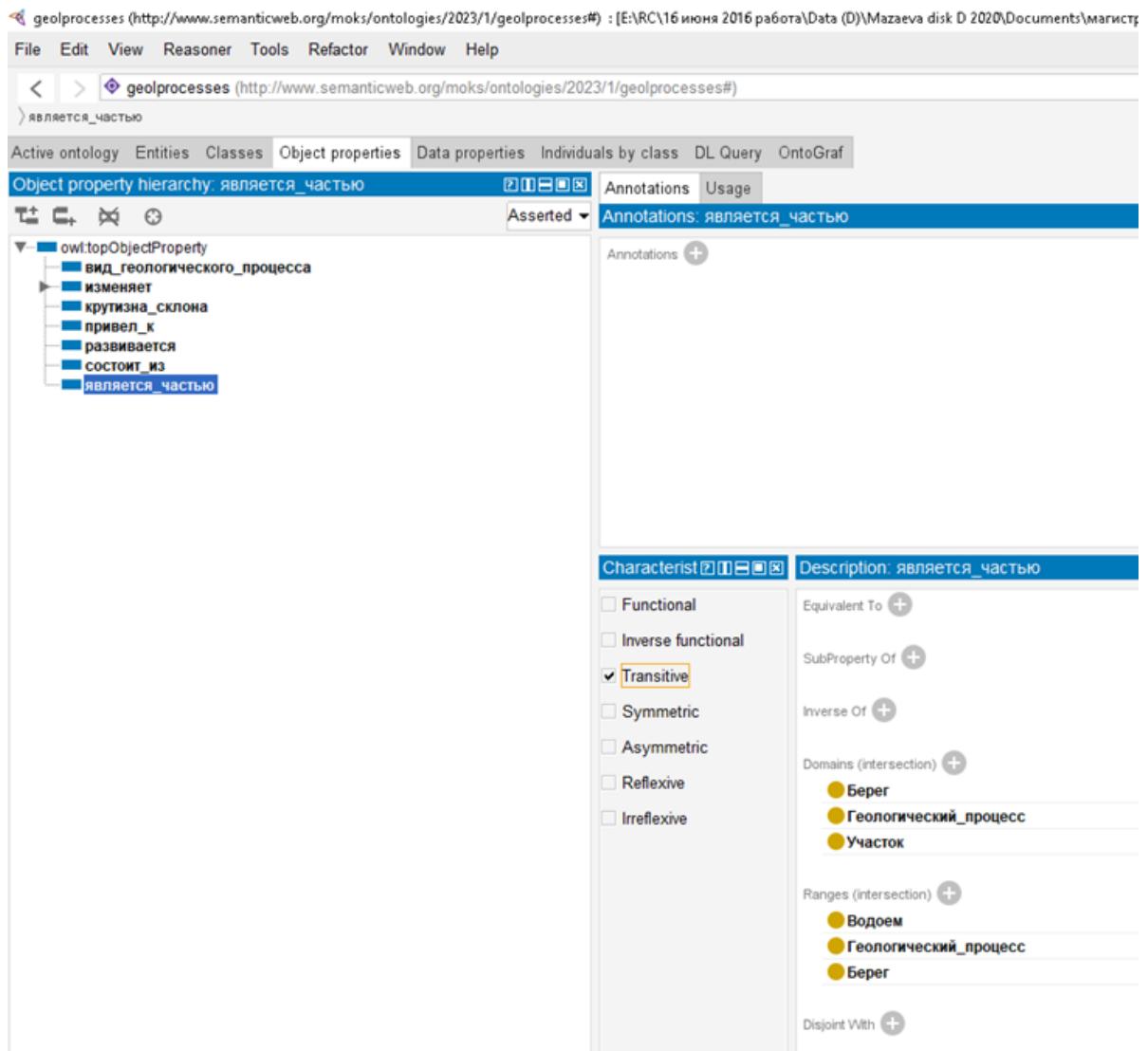


Рисунок 2.10 – Скриншот экрана панели “Object property” с примером транзитивного свойства «является частью»

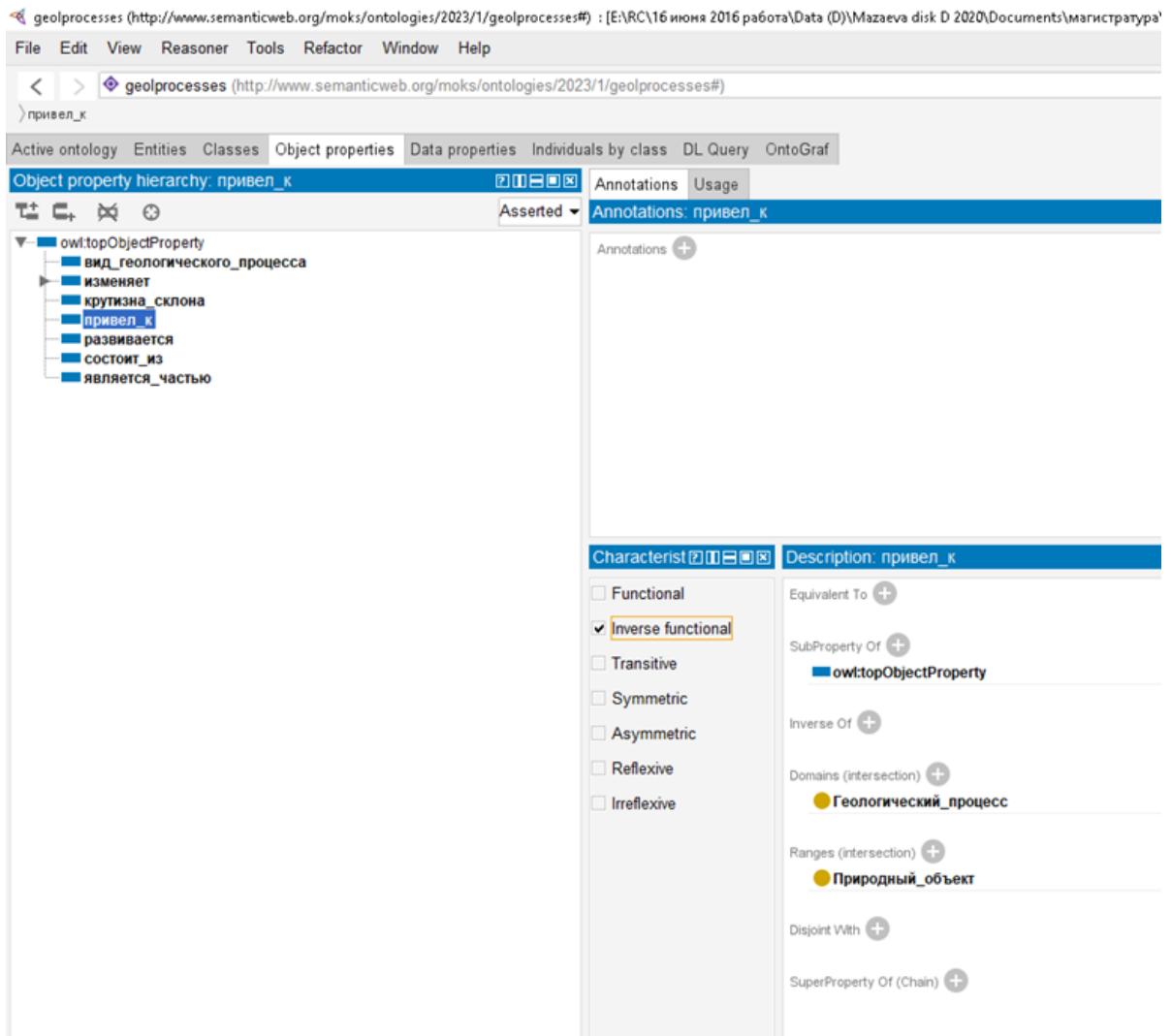


Рисунок 2.11 – Скриншот экрана панели “Object property” с примером обратно функционального свойства «привел к»

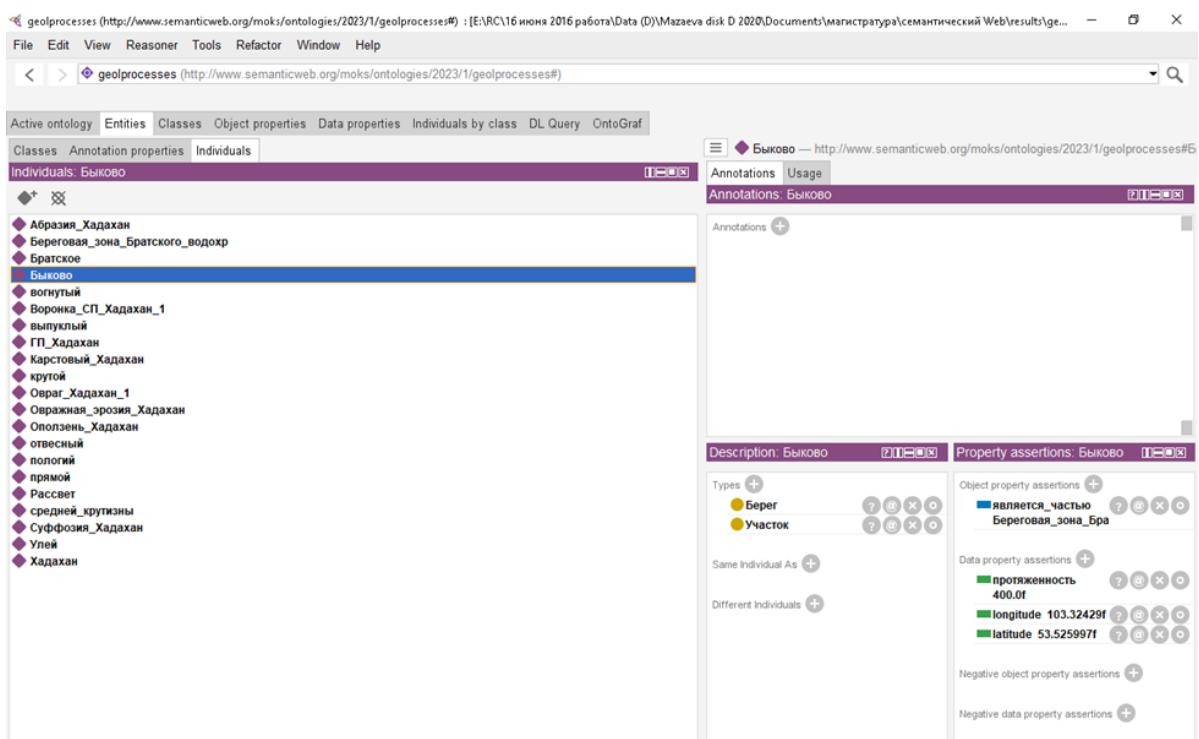


Рисунок 2.12 – Скриншот экрана с вкладкой “Individuals” для всей онтологии с описанием data property для individuals «Быково»

2.1.1 Этапы моделирования с использованием DSAS

Процесс получения результатов прогнозирования с использованием DSAS [26] состоит из следующих этапов:

1. Оцифровка исходного материала,
2. Представление контуров линий в Shape-файле специального формата,
3. Задание опорной кривой.
4. Передача данных в DSAS (ArcGIS), настройка модуля (шаг трансектов и т.п.).
5. Анализ результатов построения аппроксимации/интерполяции.

Подробно процесс представлен в разделе ?? и рисунке 2.14.

Самой трудоемкой задачей является оконтуривание береговой линии. Как правило, оно делается вручную по снимку (аэро-, спутниковому или ортофотоплану), привязанному к системе координат. Ручное оконтуривание делается при помощи манипулятора “мышь” или графического планшета, в результате получается векторный слой, Shape-файл, контур из которого, потом переносится в набор исходных контуров для DSAS.

В проекте предложено автоматизировать эту операцию при помощи использования современных средств обработки изображений на основе распознавания образов, наборов точек, относящихся

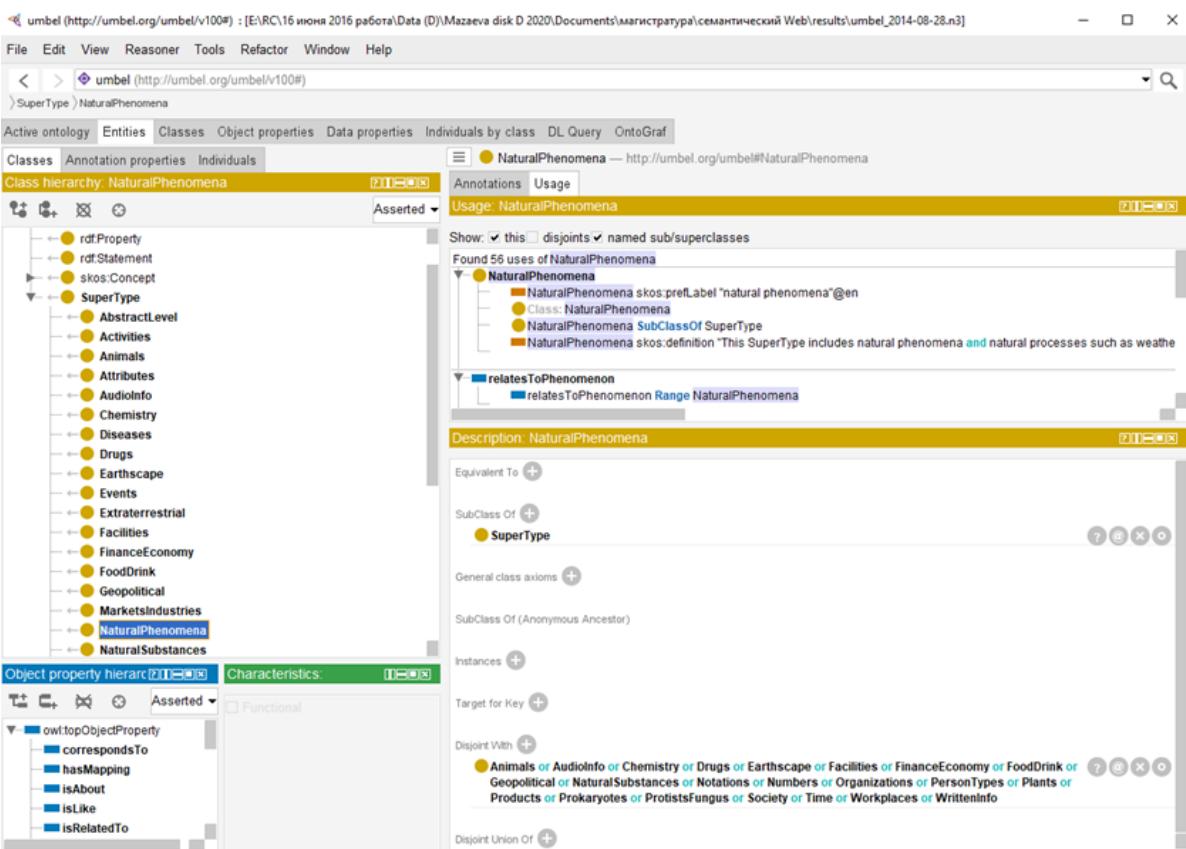


Рисунок 2.13 – Скриншот экрана существующей онтологии URI <http://umbel.org/umbel>, описывающая Natural Phenomena

к одному объекту. Теперь для получения набора контуров для одной области интереса пользователь сначала в ГИС QGIS выбирает эту область, запускает модуль `shore_qgis_module`, указывает предварительно созданную в проекте пустую группу. В модуле указывается интервал лет, за которые следует проводить распознавания контура, масштаб снимков спутника, адрес спутниковых данных.

Далее модуль запрашивает снимки из хранилищ спутниковых данных, загружает изображения (JPG) на сервер через специальный интерфейс (REST), передающий двоичные данные, принятые сервером изображение раскодируется при помощи библиотеки OpenCV. Изображения сохраняются в виде набора матриц пикселей (RGB) в группе HDF5, соответствующей имени изображения и дате. После сохранения изображения клиент сервера (QGIS-модуль `shore_qgis_module`) запускает процедуру распознавания и периодически опрашивает процессы на сервере на факт их окончания.

Процесс распознавания (SegAny), реализованный на основе Segment Anything, достаточно длительный и требующий достаточно большого объема памяти и вычислительных ресурсов. Результат распознавания помещается в подгруппу группы, хранящей изображение. Результат распознавания представляется как набор масок, обозначающих распознанные объекты, охарактеризованных атрибутами.

Маска - это матрица элементов 0 и 1, где 1 обозначает принадлежность пикселя к распознанному объекту, 0 - нет. Маски не содержат семантической информации, т.е. в данных не указано, что за объект был распознан. Атрибуты масок описывают координату ключевой точки объекта, опоясывающей прямоугольник, площадь маски в пикселях и др.

Для определения контура береговой линии необходимо из набора масок построить две граничащих друг с другом области, где границей выступает береговая линия. Для этого надо определить, какие распознанные объекты являются изображениями области, занятой водой и побережьем (ShRec). Семантику объекта можно определить при помощи запроса к данным OpenstreetMap [?]. Для этого делается запрос - “какой объект находится по этой координате”. Если пришел пустой ответ, то запрашивается “Какой объект является наиболее близкий к этой точке”. В запросе указывается тип объекта - элемент рельефа. Все маски, принадлежащие побережью и прибрежной зоне объединяются в одну.

На последнем этапе распознавания нужно построить контур. Для этого используется разработанный в [4] алгоритм, преобразующий набор пикселей в векторные данные. Результат записывается в Shape-файл, расположенный в группе, указанной пользователем на начальном этапе. Процедура распознавания SegmentAnything вносит неопределенность в геометрические параметры контура, поэтому пользователь должен проконтролировать результат визуально.

Полученный файл вручную переносится в ArcGIS для применения к ним процедуры DSAS, получение прогноза и интерпретации результатов.

2.2 Проектирование информационно-вычислительной инфраструктуры

Проектирование распределенной вычислительной инфраструктуры является трудоемким процессом, где по мимо реализации алгоритмов конкретных компонент, решаются также вопросы

передачи данных между модулями, синхронизация их совместного функционирования. В вкрапления за время производственной практики разработаны несколько модулей ИВИ, организован процесс их синхронизации с встроенным в QGIS модулем управления.

2.2.1 Функциональные требования

Совместное использование ГИС и ИВИ предполагает многопользовательскую среду, где разные ГИС делают запросы к сервисам ИВИ на выполнение некоторых задач, требующих больших вычислительных ресурсов, или специальных сервисов. Это включает также обеспечение доступа к данным из разных ГИС (QGIS, ArcGIS и др.). Современные ГИС позволяют обеспечивать доступ пространственно-распределенным данным, представленным в специальных форматах, данным баз реляционных данных, а также виртуальных географических сервисов.

2.2.2 Вычислительные процессы в ИВИ

В спроектированном распределенном программном комплексе реализуются следующие основные процессы:

1. Накопление исходных данных о береговой зоне водохранилища из следующих источников:

Векторные данные ГИС, получаемые в результате ручной оцифровки или загрузки из топоосновы, привязанные к координатам,

Полевые исследования в виде координат точек (x, y, z) относительно реперной точки с известными координатами широты и долготы;

Спутниковые снимки, получаемые из различных интернет-источников, представленные в форматах JPG, TIFF, JP2 и т.п.;

Архивные аэрофотоснимки, требующие сканирования, предварительной обработки проекции, привязки к координатам;

Ортофотоплан, формируемый сканированием местности при помощи квадрокоптера, сшивания отдельных снимков, коррекции проекции;

2. Оцифровка исходных данных с целью получения данных типа 1;

Ручная оцифровка – исследователь сам анализирует снимок и формирует векторные данные при помощи инструментов ГИС;

Автоматизированная оцифровка – некоторые ГИС позволяют «обводить» контуры объектов, ориентируясь на начальную точку контура, указанную человеком, и анализируя разницу цветовых характеристик контура;

Автоматическая оцифровка, позволяющая в автоматическом режиме находить на изображении контуры береговой линии (без значительного участия человека);

3. Подготовка входных данных для построения моделей измерения контура и других характеристик береговой линии;
4. Вычисление прогноза, и его представление в виде данных ГИС;
5. Визуализация результатов моделирования и мониторинга;
6. Интерпретация результатов:

Неавтоматический анализ, где результаты моделирования анализируются специалистом «вручную» и вырабатываются рекомендации по безопасному использованию береговой зоны;

Автоматизированный анализ, где часть анализа результата передана экспертной системе.

На рисунке 2.14 изображены основные описанные выше процессы.

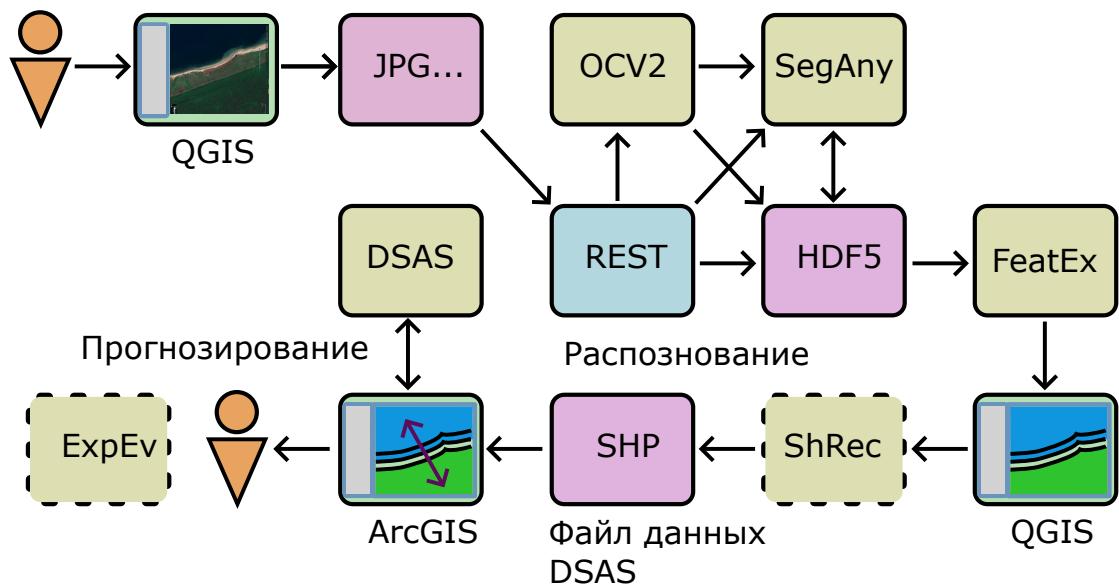


Рисунок 2.14 – Основные процессы в ИВИ поддержки исследований береговых зон водохранилищ

2.2.3 Архитектура системы

Перечисленные в предыдущем разделе функции и взаимодействия между ними формируют собой элементы архитектуры распределенной системы обработки ГИС-данных (рисунок 2.15).

Система состоит из двух основных частей, обычно обозначаемых как «клиент» и «сервер». В нашем случае они взаимодействуют друг с другом по протоколу HTTP и реализуют взаимодействие через интерфейс «архитектуры REST» (Representation State Protocol). QGIS и ArcGIS - это

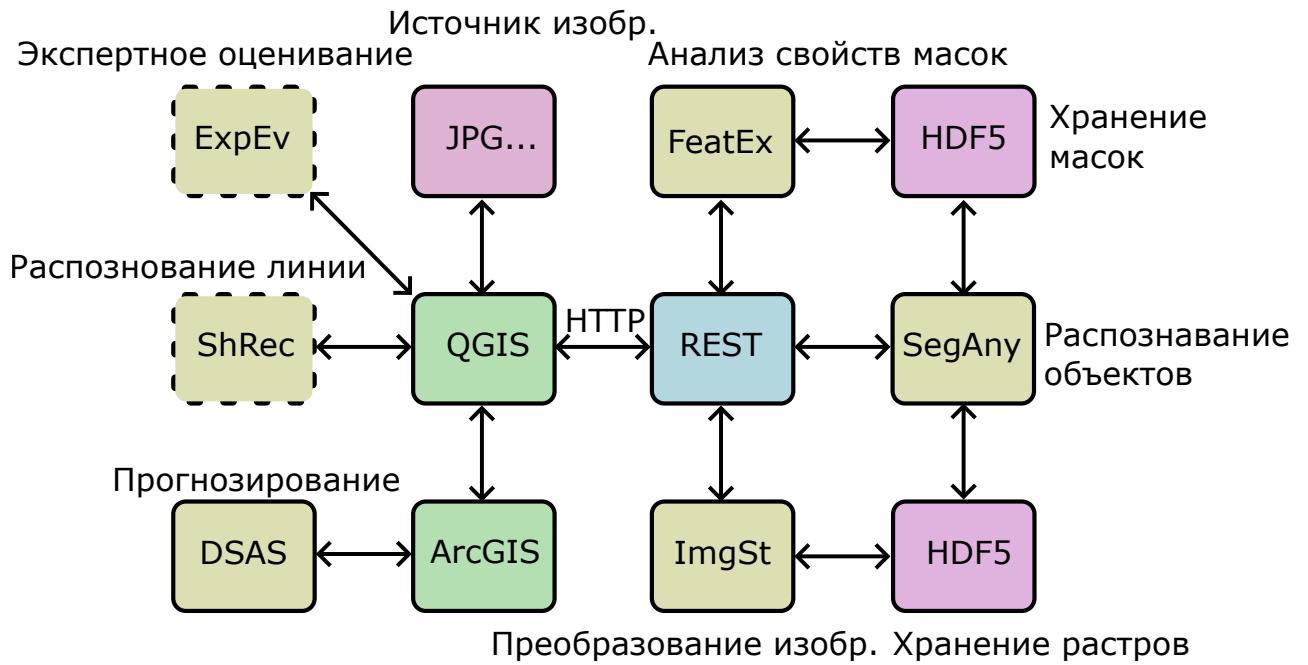


Рисунок 2.15 – Архитектура распределенной системы

клиентская часть, она включает также процедуры DSAS, ShRec (распознавание контура), ExpEv (экспертная оценка). QGIS также выступает клиентом для сервера изображений (формат JPG, TIFF, JP2 и др.), получаемых, в том числе, с сервера Google Maps.

Серверная часть – набор модулей, запускаемых удаленно из модуля `shore_qgis_module`, т.е. процедурами распознавания управляет полностью клиентская часть. Управление осуществляется через интерфейс REST, где в качестве объектов выступают сервисы хранения исходных изображений (OCV2 + HDF5), распознавания объектов на изображении (SegAny), анализ масок и изображений с целью получения дополнительных характеристик (FeatEx), сервис выгрузки данных с представлением в JSON (внутри REST).

2.2.4 Распознавание контура береговой линии

Как было сказано в разделе, посвященном перечню требований к ИВИ, задача распознавания контура береговой линии решается в условиях ограниченных вычислительных ресурсов и отсутствия предварительно обработанной информации, пригодной для использования современными алгоритмами машинного обучения. Не смотря на данную ситуацию есть возможность применения в задачах распознавания предобученных моделей машинного обучения.

2.2.4.1 Библиотека Segment Anything

SAM (Segment Anything Model) — это сегментационная модель, которая была выпущена Meta AI* весной 2023 года и быстро стала одной из самых популярных AI-моделей. SAM называют первой фундаментальной моделью в компьютерном зрении и сравнивают с ChatGPT в NLP из-за рекордно большого количества разнообразных данных, которые видела модель; а также из-за её способности к zero-shot transfer, то есть способности легко обобщаться для решения смежных задач.

SAM обучается на наборе данных Segment Anything 1-Billion mask (SA-1B), который содержит набор из 11 миллионов изображений и более 1 миллиарда масок. Это делает модель очень надежной при определении границ объектов и различении различных объектов в разных доменах, даже если модель раньше с такими объектами не сталкивалась ??.

SAM сегментирует объекты на картинке в соответствии с промптом: им может быть точка на изображении, приблизительный прямоугольник или произвольный текст. Авторы отмечают, что при заранее вычисленном эмбеддинге изображения модель способна работать на CPU в браузере в реальном времени, а также может применяться без дообучения для задач, отличных от сегментации (например, для задач детекции границ и положения объектов — edge detection и object proposal).

Кодер изображений реализован в PyTorch, и для эффективного вывода требуется графический процессор. Кодер подсказок и декодер маски могут работать непосредственно с PyTorch или конвертироваться в ONNX и эффективно работать на ЦП или графическом процессоре на различных платформах, поддерживающих среду выполнения ONNX [33].

2.2.5 Анализ структуры объектов изображения

На данном этапе производится классификация объектов – необходимо соотнести распознанные объекты к классам «водная поверхность» и «суша». Модель SA для каждого распознанного объекта сопоставляет так называемую ключевую точку (рисунок 2.16). В режиме распознавания «всех объектов» на изображении библиотека сканирует изображения по некоторой сетке. Каждой точкой сетки производится указание (prompt) на объект, который необходимо выделить на изображении. Если указанию соответствует новый объект, то этот объект ассоциируется с точкой. Получается, что координаты ключевой точки идентифицируют объект.

На рисунке 2.16 точки A, B, C, D обозначают периметр изображения, координаты которого используется для определения свойств объектов (например, как касающихся границы), объекты E, F, G – собственно некоторые распознанные объекты. Кружочками обозначены ключевые точки, координаты которых идентифицируют объекты. Таким образом, объекты F, G, и объект, идентифицируемый K – это берег водохранилища, содержащий нужный контур a'–b'. Кривая a–b – граница водохранилища, полученная с OpenStreetMap, ее можно использовать как «начальное приближение» или опорный объект выделения множества граничных точек объединенного контура берега. Кривая c–d (дорога вдоль границы леса) принадлежит побережью и находится дальше от ab, чем a'–b' и не может быть побережьем. Дополнительные свойства объектов, используемые для фильтрации получаются запуском процедуры совместного анализа данных масок (FeatEx) и

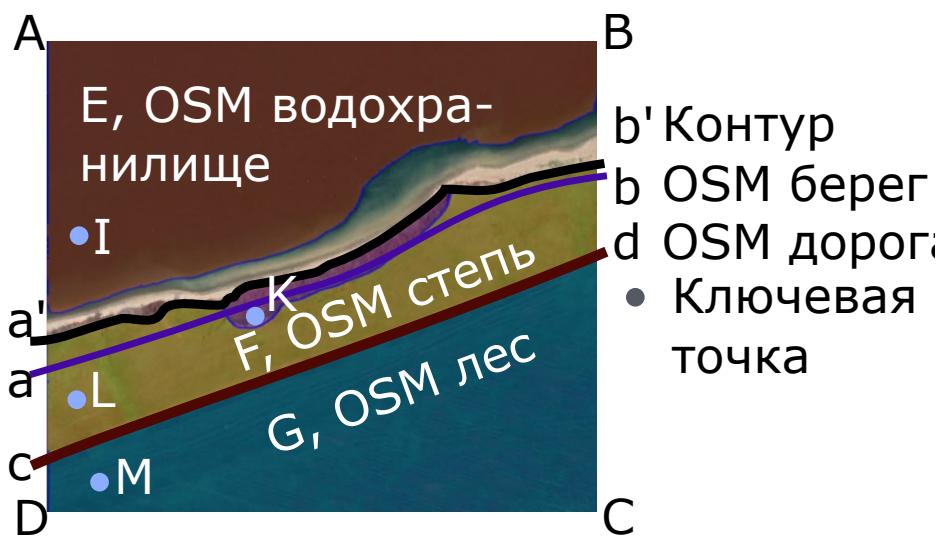


Рисунок 2.16 – К объяснению идеи распознавания береговой линии при помощи классификации масок SsegmentAnything

изображения с целью порождения дополнительных атрибутов объектов, например, средний цвет объекта, расстояние границы объекта к границе изображения, общая граница с другими объектами и т.п.

Точка, таким образом, принадлежит объекту, точнее находится на маске изображения. Чтобы достаточно достоверно определить, является объект водным или сушей, можно выполнить запрос к серверу OpenStreetMap [5] следующего вида:

```

1 [out:json];
2 (
3     is_in(53.5202071,103.3259833); // Координата точки К
4 );
5 out center;

```

В ответ будет получен JSON следующего вида:

```

1 {
2     "version": 0.6,
3     "generator": "Overpass API 0.7.62.1 084b4234",
4     "osm3s": {
5         "timestamp_osm_base": "2024-06-12T10:11:58Z",
6         "timestamp_areas_base": "2024-06-12T05:25:19Z",
7         "copyright": "The data included in this document is from www.openstreetmap.org. The
8             data is made available under ODbL."
9     },

```

```

9   "elements": [
10
11   {
12     "type": "area",
13     "id": 3600060189,
14     "tags": {
15       "IS03166-1": "RU",
16       "IS03166-1:alpha2": "RU",
17       "IS03166-1:alpha3": "RUS",
18       "IS03166-1:numeric": "643",
19       "according_to:RU": "yes",
20       "according_to:UA": "no",
21       "admin_level": "2",
22       "alt_name:eo": "Rusujo;Ruslando",
23       "border_type": "nation",
24       "boundary": "administrative",
25       "currency": "RUB",
26       "default_language": "ru",
27       "flag": "https://upload.wikimedia.org/wikipedia/commons/f/f3/Flag_of_Russia.svg",
28       "int_name": "Russia",
29       "int_ref": "RU",
30       "name": "Россия",
31       // ....
32     },
33     // ....
34   {
35     "type": "area",
36     "id": 3601221148,
37     "tags": {
38       "addr:country": "RU",
39       "admin_level": "3",
40       "boundary": "administrative",
41       // ....
42     }

```

```
43 // ....
```

Среди перечисленных объектов нет объекта, обозначающего водный резервуар. Теперь запрос по координате точки *I*:

```
1 [out:json];
2 (
3     is_in(53.5202071,103.3259833); // Координата точки I
4 );
5 out center;
```

В ответ будет получен JSON следующего вида:

```
1 // .....
2 {
3     "type": "area",
4     "id": 3600167043,
5     "tags": {
6         "alt_name:cs": "Bratsk\u0103 vodn\u0103 n\u00e1dr\u00f9z",
7         "gvr:code": "16010100821416200001042",
8         "name": "\u0411\u0430\u0434\u0430\u043d\u0430\u043b\u043e\u0436\u0435\u043d\u0438\u044f \u0432\u043e\u0434\u043e\u0431\u0438\u043d\u0430\u0438\u0437\u0430\u043d\u0430\u043b\u0438\u0435",
9         "name:ca": "Embassament de Bratsk",
10        "name:cs": "Bratsk\u0103 p\u00e9chradn\u0103 n\u00e1dr\u00f9z",
11        "name:en": "Bratsk Reservoir",
12        "name:eo": "Bratska Rezervujo",
13        "name:pl": "Zbiornik Bracki",
14        "name:ru": "\u0411\u0430\u0434\u0430\u043d\u0430\u043b\u043e\u0436\u0435\u043d\u0438\u044f \u0432\u043e\u0434\u043e\u0431\u0438\u043d\u0430\u0438\u0437\u0430\u043d\u0430\u043b\u0438\u0435",
15        "name:sk": "Bratsk\u0103 vodn\u0103 n\u00e1dr\u00f9z",
16        "name:zh": "\u0411\u0430\u0434\u0430\u043d\u0430\u043b\u043e\u0436\u0435\u043d\u0438\u044f \u0432\u043e\u0434\u043e\u0431\u0438\u043d\u0430\u0438\u0437\u0430\u043d\u0430\u043b\u0438\u0435",
17        "natural": "water",
18        "type": "multipolygon",
19        "water": "reservoir", // Тег, определяющий тип водема
20        "wikidata": "Q899803",
21        "wikipedia": "ru:Братское водохранилище"
22    }
23 },
24 // .....
```

Здесь среди выданных объектов присутствует объект, описанный как водный резервуар.

Собрав информацию по всем ключевым точкам, теперь становится возможным объединить все маски, относящиеся к водным объектам, и все маски, относящиеся к суше. Граница суши со стороны водного объекта будет являться приближением береговой линии. Контуры береговой линии $a'-b'$ из набора пикселей (растра) преобразуются в векторный объект при помощи варианта алгоритма, представленного в [4].

Предлагаемый подход также применим к привязанным к координатам аэрофотоснимкам и ортофотопланам.

3 Реализация подсистем распределенной вычислительной среды

За время работы над задачей реализации ГИС береговой зоны разработаны некоторые модули подсистемы, изображенные на рисунке 2.15 непрерывными линиями.

3.1 Модуль управления процессом моделирования

Данный модуль реализован в виде расширения QGIS. Модули расширения реализуются согласно стандартизованной технологии включающей

- интерфейс пользователя, проектируемый при помощи QT-дизайнера,
- класс, реализующий конкретные функции.

Интерфейс пользователя, реализованный в модуле изображен на рисунке 3.1. Управление модулем производится при помощи панели в трее. Интерфейс включает четыре вкладки: выбор целевой группы слоев, в которую будет помещаться результат распознавания в виде shape-файла; Выбор тайл-слоя, откуда загружается изображение для анализа; перечень сгенерированных shape-слоев; настройка модуля, в частности, адрес сервера хранения и обработки изображений.

Задача модуля – координировать процесс обработки изображений. Взаимодействие модуля с сервером осуществляется через интерфейс (модуль) REST.

3.2 Интерфейс REST загрузки и обработки изображений

REST (Representational state transfer) – это стиль архитектуры программного обеспечения для распределенных систем, таких как World Wide Web, который, как правило, используется для построения веб-служб. Термин REST был введен в 2000 году Роем Филдингом, одним из авторов HTTP-протокола. Системы, поддерживающие REST, называются RESTful-системами.

В общем случае REST является очень простым интерфейсом управления информацией без использования каких-то дополнительных внутренних прослоек. Каждая единица информации однозначно определяется глобальным идентификатором, таким как URL. Каждая URL в свою очередь имеет строгий заданный формат.

Задача интерфейса REST - принимать бинарный файл изображения, преобразовать его в матрицы слоев, записать в базу данных на сервере, выдать глобальный идентификатор, при помощи которого далее идентифицируется сохраненные данные изображения, запуск других подсистем. Приведем пример реализации интерфейсов REST для загрузки изображений и управления процессом распознавания.

```
1 img = Service(name='imgstore',
2                 path='/sa-1.0/image/{img_name}',
```

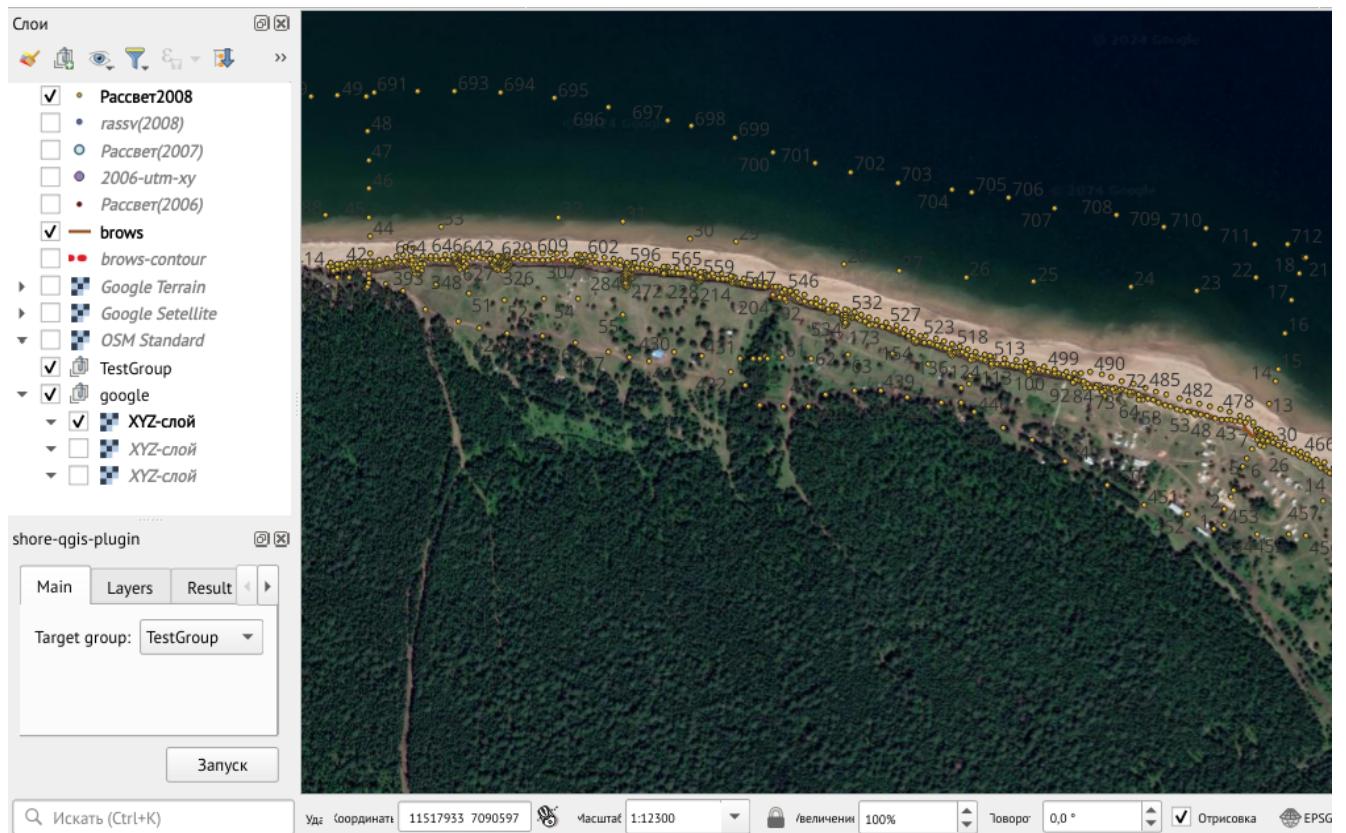


Рисунок 3.1 – Интерфейс пользователя, реализованный в модуле расширения

```

3      description="Image collection")
4
5 @img.put()
6 def put_image(request):
7     """Принимает бинарные данные картинки,
8     возвращает JSON с UUID сохраненного изображения
9 """
10    name = request.matchdict['img_name']
11
12    imgg, ds = add_image(name, request.body)
13
14    uui = uuidgen()
15    uuis = str(uui)
16    pth = ds.name
17    STORAGE, INGRP, UUIDGRP = storage_begin()
18    if name in UUIDGRP:

```

```
19 ouuis = UUIDGRP[name]
20
21     del UUIDGRP[name]
22
23     del UUIDGRP[ouuis]
24
25     # Отображение UUID <-> имя изображения
26
27     UUIDGRP.create_dataset(name, data=uuis)
28
29     UUIDGRP.create_dataset(uuis, data=name)
30
31     STORAGE, INGRP, UUIDGRP = storage_end()
32
33     return {
34
35         "error": None,
36
37         "ok": True,
38
39         "uuid": uuis,
40
41         "content": pth,
42
43         "name": name,
44
45         "namepath": imgg.name
46
47     }
48
49
50
51 def add_image(name, content, replace=True):
52     """Добавление изображения в БД"""
53
54     nparr = np.frombuffer(content, np.uint8)
55
56     image = cv2.imdecode(nparr, cv2.IMREAD_COLOR)
57
58     image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
59
60     # Открытие БД
61
62     STORAGE, INGRP, UUIDGRP = storage_begin()
63
64     if name in INGRP:
65
66         del INGRP[name]
67
68         imgg = INGRP.create_group(name)
69
70         ds = imgg.create_dataset('content',
71
72             data=image, compression="lzf")
73
74         log.info("Image '{}' loaded".format(name))
75
76     # Закрытие БД
77
78     STORAGE, INGRP, UUIDGRP = storage_end()
79
80
81     return (imgg, ds)
```

Загрузка изображения делается при помощи запроса PUT с адресом `https://<адрес сервера>:651.0/image/<имя картинки>`. В теле запроса PUT помещается содержимое изображения. В качестве

результата возвращается JSON-ответ, содержащий в поле «`uuid`» идентификатор сохраненного изображения. Список сохраненных изображений получается запуском запроса GET к этому же адресу без указания имени изображения. Возвращается JSON со списком отображений UUID-<имя файла изображения>. Полученный идентификатор используется для обозначения изображения, подвергающегося процедуре распознавания:

```
1  sactrl = Service(name='segment-any-control',
2                      path='/sa-1.0/sa/{img_uuid}/{cmd}',
3                      description="Functions of SA on a image\
4                                     identified by uuid")
5  @sactrl.post()
6  def start_recognition(request):
7      # Импорт процедур анализа, выполняющихся в
8      # других процессах
9      from ..tasks import (sa_start,
10                         ANSWERS, rc_set, rc_get, rc_delete,
11                         rc_update)
12
13      uuids = request.matchdict['img_uuid']
14      cmd = request.matchdict['cmd']
15      # cmd = 'start'
16
17      STORAGE, INGRP, UUIDGRP = storage_begin()
18      # идентификатор изображения существует?
19      isimg = uuids in UUIDGRP
20      STORAGE, INGRP, UUIDGRP = storage_end()
21
22      rd = {"error": False, "ok": True, "cmd": cmd}
23
24      if cmd == "flush":
25          ANSWERS.flushdb()
26          return rd
27
28      if not isimg:
29          return {
```

```

30         "error": "not found",
31         "ok": False,
32         "uuid": uuids,
33         "cmd": cmd,
34         "processuuid": None
35     }
36
37     # Команда запуска процесса распознавания
38     if cmd == "start":
39
40         prevrc = rc_get(uuids)
41
42         if prevrc is not None:
43
44             return {
45                 "error": "already running",
46                 "ok": False,
47                 "uuid": uuids,
48                 "cmd": cmd,
49                 "processuuid": prevrc.get("processuuid",
50                                         None),
51                 "ready": prevrc.get("ready", False)
52             }
53
54             del prevrc
55
56             rc = {"uuid": uuids, "ready": False}
57
58             rc_set(uuids, rc)
59
60             arc = sa_start.delay(uuids)
61
62             puuid = str(arc.id)
63
64
65             def _u(r):
66
67                 r["processuuid"] = puuid
68
69
70                 rc = rc_update(uuids, _u)
71
72                 print(rc_get(uuids))
73
74                 puuid = rd["processuuid"] = puuid
75
76
77                 # Команда проверки завершения процесса
78
79                 elif cmd == "status":
80
81                     rd["ready"] = False

```

```

64
65     def _a(v, rr):
66         rd["ready"] = v
67         rd["result"] = rr.get("result", None)
68
69         rc = rc_get(uuids, "ready", _a)
70
71         if rc is None:
72             return {
73                 "error": "no process",
74                 "ok": False,
75                 "uuid": uuids,
76                 "cmd": cmd,
77                 "ready": None
78             }
79
80         rd["processuuid"] = rc["processuuid"]
81
82         # Команда завершения процесса и удаления его данных
83
84         elif cmd == "finalize":
85             rcg = rc_get(uuids)
86
87             if rcg is None:
88                 return {
89                     "error": "not running",
90                     "ok": False,
91                     "uuid": uuids,
92                     "cmd": cmd,
93                     "ready": None
94                 }
95
96             rc_delete(uuids)
97
98             rd.update({"ready": rcg["ready"],
99                         "processuuid": rcg["processuuid"]})
100
101         # Команда отмены процесса
102
103         elif cmd == "discard":
104             rcg = rc_get(uuids)
105
106             if rcg is not None:

```

```

98     return {
99         "error": "still running",
100        "description":
101            "cannot stop SA, wait its finishing. \
102                Use status command.",
103        "ok": False,
104        "uuid": uuids,
105        "cmd": cmd,
106        "ready": None
107    }
108    STORAGE, INGRP, UUIDGRP = storage_begin()
109    name = gs(UUIDGRP[uuids])
110    imgg = INGRP[name]
111    if "masks" in imgg:
112        del imgg["masks"]
113        rc = "removed"
114    else:
115        rc = "no mask"
116    STORAGE, INGRP, UUIDGRP = storage_end()
117    rd.update({
118        "ready": None,
119        "processuuid": None,
120        "description": rc
121    })
122
123    return rd

```

Запуск процесса распознавания осуществляется при помощи запроса POST следующего формата - `http://<адрес сервера>:6543/sa-1.0/sa/<идентификатор изображения>/<команда>`. Командой выступает ключевое слово - одно из четырех:

start – запуск нового процесса,

status – запрос состояния распознавания,

finalize – завершение исполнившегося процесса,

flush – отмена процесса.

Требующие большие вычислительные ресурсы процессы выполняются в отдельных процессах сервера или на специализированных серверах, снабженных аппаратной поддержкой вычислений (CUDA и ему подобным). Реализация запуска таких задач реализована при помощи библиотеки celery языка программирования Python.

3.3 Исполнение задач с долгим временем вычисления

На этапе предварительного ознакомления с библиотекой SegmentAnything было установлено, что среднее время выполнения распознавания объектов на изображении, взятом с Google-сервера - 3.5 минуты на персональном компьютере, оборудованном ОС Linux, видеокартой NVIDIA 1030 и использование библиотеки tensorflow с поддержкой CUDA. Если операцию напосредственно выполнять из интерфейса пользователя, то работа с ГИС QGIS для пользователя будет некомфортной – он должен будет ждать результата, но при этом не будет возможности выполнять какие-либо другие операции. Для решения этой проблемы и был разработан сервер хранения и обработки изображений. Пользователи задают задачи «длительного выполнения», задачи ставятся в очередь на сервере, по завершению их выполнения модуль QGIS запрашивает результат и помещает его в проект.

Реализация очереди выполнена при помощи отдельного объекта REST и процесса - сервера очередей celery (сельдерей). Работа очереди обеспечивается еще одним дополнительным сервисом – RabbitMQ – «Rabbit» message queue, ранее установленным в сети ИДСТУ как раз для поддержки очередей. RabbitMQ позволяет синхронизировать вычислительные процессы на основе организации передачи сообщений, а также распределять нагрузку между вычислительными узлами, что дает возможность разрабатывать масштабируемые ИВИ.

3.4 Хранение изображений

Загруженное через REST-интерфейс изображение декодируется и преобразуется в набор матриц (тензор), соответствующий слоям Red-Green-Blue-Alpha. Преобразование осуществляется перед выполнением сохранения. Такой подход позволяет на этапе передачи изображения на обработку загружать данные непосредственно из хранилища в модули обработки в виде объектов numpy. Идея реализована при помощи файлов формата HDF5.

3.4.1 Стандарт HDF5: формат и инструментарий

Иерархический формат данных версии 5 (HDF5) - это формат файлов с открытым исходным кодом, который поддерживает большие, сложные и неоднородные данные. HDF5 использует структуру типа ”файловый каталог которая позволяет организовывать данные внутри файла множеством способов, как это делается в папках на вашем компьютере. Формат HDF5 также позволяет встраивать метаданные, делая его самоопределяющимся.

HDF5 — современная версия формата. Получил премию R&D100 от журнала «R&D Magazine» в 2002 году []. Содержит иерархию из двух основных типов объектов:

Datasets — наборы данных, многомерные массивы объектов одного типа

Groups — группы, являются контейнерами для наборов данных и других групп

Содержимое файлов HDF5 организовано подобно иерархической файловой системе, и для доступа к данным применяются пути, сходные с POSIX-синтаксисом, например, /path/to/resource. Метаданные хранятся в виде набора именованных атрибутов объектов (рисунок ??).

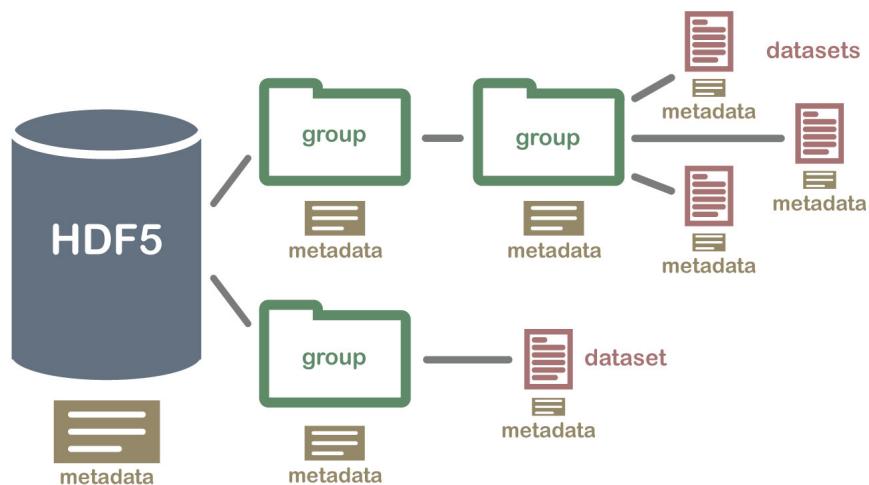


Рисунок 3.2 – Вариант хранения информации в формате HDF5

3.4.2 Хранение изображений и масок

Изображения, преобразованные в тензоры, сохраняются в корневой узел хранения изображений в виде группы, поименованной именем файла изображения. Маски, распознанные на изображении помещаются в подгруппу «masks». Тензор изображения помещается в узел «content». Генерированный идентификатор (UUID) помещается в атрибут группы изображения, а также в специальный узел отображения идентификаторов в имена изображений и обратно. Тензоры изображений сжимаются алгоритмом LZW, использование которого позволяет загружать тензор прямым проходом по данным. Маски также содержат узел «content», к которому привязаны атрибуты, ассоциированными с маской.

Доступ к данным изображения, масок, атрибутам и т.п. реализован также в интерфейсе REST. В частности, маски выгружаются по порядковым номерам и идентификаторам соответствующего изображения.

3.5 Тестирование

К текущему моменту реализованы не все модули представленной в ВКР архитектуры, поэтому подемонстрируем только работоспособное взаимодействие пользователя, модуля QGIS, сервера обработки изображений. Вычислительный процесс начинается с момента нажатия кнопки «Запуск» в интерфейсе модуля QGIS. До этого, как было сказано выше, пользователь должен создать проект, подключить XYZ Tile Layer, например, Google Maps, поставляющий растровые данные спутниковых изображений. Также необходимо установить интересующий регион при помощи манипуляции с картами и визуальным контролем.

Модуль получает изображения Tile-слоя в виде JPG-изображения и посылает его на сервер при помощи запроса PUT (для наглядности мы моделируем запрос при помощи программы curl):

```
$ curl -X PUT http://127.0.0.1:6543/sa-1.0/image/Bykovo.jpg --data-binary "@Bykovo6.JPG"
{"error": null, "ok": true, "uuid": "a535ef20-2b7b-11ef-9f92-704d7b84fd9f", "content": "..."}
```

В результате получен ответ: операция проведена успешно, изображение помещено в хранилище в виде тензора под идентификатором «a535ef20-2b7b-11ef-9f92-704d7b84fd9f». Далее данный идентификатор будет использоваться для указания тензора. На запрос сервер отреагировал записью в журнале:

```
Starting server in PID 14014.
```

```
2024-05-16 08:49:41,197 INFO [waitress:485][MainThread] Serving on http://127.0.0.1:6543
2024-05-16 08:49:41,198 INFO [waitress:485][MainThread] Serving on http://[::1]:6543
2024-05-16 08:59:12,365 INFO [shores_server.views.rest:44][waitress-1] Successfully o
2024-05-16 08:59:12,427 INFO [shores_server.views.rest:149][waitress-1] Image 'Bykovo
2024-05-16 08:59:12,453 INFO [shores_server.views.rest:44][waitress-1] Successfully o
```

Далее запускаем процесс распознавания объектов, передавая идентификатор тензора и указывая команду start:

```
$ curl -X POST http://127.0.0.1:6543/sa/a535ef20-2b7b-11ef-9f92-704d7b84fd9f/start
{"error": false, "ok": true, "cmd": "start", "processuuid": "bc31a029-8cc6-4c9d-803f-7f46d2b10def"}
```

Возвращено сообщение о том, что процесс запущен и идентификатор этого процесса.

Журнал REST-модуля дополняется записью

```
2024-05-16 08:59:12,427 INFO [shores_server.views.rest:149][waitress-1] Image 'Bykovo
2024-05-16 08:59:12,453 INFO [shores_server.views.rest:44][waitress-1] Successfully o
2024-05-16 09:04:18,513 INFO [shores_server.views.rest:44][waitress-3] Successfully o
```

```
{'uuid': 'a535ef20-2b7b-11ef-9f92-704d7b84fd9f', 'ready': False, 'processuuid': 'bc31a8cc6-4c9d-803f-7f46d2b10def'}
```

Сервер celery реагирует на сообщение

```
----- celery@center.irnok.net v5.3.6 (emerald-rush)
--- *****
-- ***** Linux-6.6.30-1-lts-x86_64-with-glibc2.39 2024-06-16 08:49:41
- ** --- *
- ** ----- [config]
- ** ----- .> app:           __main__:0x7bc3a2b065a0
- ** ----- .> transport:    amqp://shores:**@192.168.191.131:5672/shores
- ** ----- .> results:      sqlite:///results.db
- ** --- * --- .> concurrency: 4 (prefork)
-- ***** .> task events: OFF (enable -E to monitor tasks in this worker)
--- *****
----- [queues]
     .> celery          exchange=celery(direct) key=celery
```

```
2024-05-16 08:49:43,119 WARNI [py.warnings:110][MainThread] /home/eugeneai/.pyenv/vers
server/lib/2024-05-16 08:49:43,354 INFO  [celery.worker.consumer.connection:22][MainTh
2024-05-16 08:49:43,358 WARNI [py.warnings:110][MainThread] /home/eugeneai/.pyenv/vers
server/lib/python3.12/site-packages/celery/worker/consumer/consumer.py:507: CPendingDe
whether broker connection retries are made during startup in Celery 6.0 and above.
2024-05-16 08:49:43,435 INFO  [celery.worker.consumer.mingle:40][MainThread] mingle: s
2024-05-16 08:49:44,745 INFO  [celery.worker.consumer.mingle:49][MainThread] mingle: a
2024-05-16 08:49:45,092 INFO  [celery.apps.worker:175][MainThread] celery@center.irnok
...
2024-05-16 09:04:18,804 INFO  [celery.worker.strategy:161][MainThread] Task shores_ser
8cc6-4c9d-803f-7f46d2b10def] received
2024-05-16 09:04:18,812 INFO  [shores_server.tasks:187][MainThread] creating task proc
2b7b-11ef-9f92-704d7b84fd9f
2024-05-16 09:04:18,819 INFO  [shores_server.views.rest:44][MainThread] Successfully o
```

```
2024-05-16 09:04:18,830 INFO [root:191][MainThread] Image name is 'Bykovo.jpg'  
2024-05-16 09:04:18,896 INFO [root:195][MainThread] Image shape is (979, 1568, 3)  
2024-05-16 09:04:18,897 INFO [root:203][MainThread] New recognition starting.  
2024-05-16 09:04:18,898 INFO [root:205][MainThread] Torch engine is cpu.  
2024-05-16 09:04:18,898 INFO [root:92][MainThread] SAM starts loading  
2024-05-16 09:04:25,912 INFO [root:102][MainThread] SAM loaded 'vit_b'  
2024-05-16 09:04:25,912 INFO [root:115][MainThread] Start Recognition/Segmentation  
2024-05-16 09:06:09,571 INFO [root:117][MainThread] Finish Recognition/Segmentation  
2024-05-16 09:06:09,571 INFO [root:216][MainThread] Recognition finished, saving into  
2024-05-16 09:06:09,573 INFO [shores_server.views.rest:44][MainThread] Successfully o  
2024-05-16 09:06:09,632 INFO [root:243][MainThread] Found 8 masks  
2024-05-16 09:06:09,891 INFO [celery.app.trace:131][MainThread] Task shores_server.ta  
8cc6-4c9d-803f-7f46d2b10def] succeeded in 111.08417257107794s: None
```

Сервер SegmentAnything загрузил модель, передал управление процедуре распознавания, SA распознал 8 объектов, эти объекты помещены в HDF5-хранилище в виде масок.

Маски выгружаются с сервера при помощи следующего запроса:

Сервер реагирует, в его журнал добавляется запись (факт доступа к HDF5-хранилищу):

```
2024-05-16 09:14:29,200 INFO [shores_server.views.rest:44][waitress-0] Successfully o  
2024-05-16 09:14:29,208 INFO [shores_server.views.rest:44][waitress-0] Successfully o
```

Далее можно запустить еще один процесс, процесс изъятия значений для характеристик масок и представление их в виде графа знаний, но сначала надо сообщить серверу, что мы завершили работу с модулем SA:

```
$ curl -X POST http://127.0.0.1:6543/sa-1.0/sa/a535ef20-2b7b-11ef-9f92-704d7b84fd9f/fin  
{"error": false, "ok": true, "cmd": "finalize", "ready": true, "processuuid": "bc31a02  
8cc6-4c9d-803f-7f46d2b10def"}  
# Запуск feature extraction  
$ curl -X POST http://127.0.0.1:6543/sa-1.0/fe/a535ef20-2b7b-11ef-9f92-704d7b84fd9f/st
```

```

{"error": false, "ok": true, "cmd": "start", "processuuid": "1ee19583-ae48-471f-9081-25b840510d3c"}

# Запрос статуса операции - завершен

$ curl -X POST http://127.0.0.1:6543/sa-1.0/fe/a535ef20-2b7b-11ef-9f92-704d7b84fd9f/status
{"error": false, "ok": true, "cmd": "status", "ready": true, "result": true, "processuuid": "ae48-471f-9081-25b840510d3c"}
```

Завершение задачи feature extraction

```
$ curl -X POST http://127.0.0.1:6543/sa-1.0/fe/a535ef20-2b7b-11ef-9f92-704d7b84fd9f/finish
{"error": false, "ok": true, "cmd": "finzlize"}
```

Журнал Celery:

```

2024-05-16 09:21:48,141 INFO [celery.worker.strategy:161][MainThread] Task shores_server-ae48-471f-9081-25b840510d3c received
2024-05-16 09:21:48,144 INFO [shores_server.tasks:268][MainThread] Creating task process-2b7b-11ef-9f92-704d7b84fd9f
2024-05-16 09:21:48,144 INFO [shores_server.views.rest:44][MainThread] Successfully opened connection to DB
2024-05-16 09:21:48,228 INFO [root:300][MainThread] Starting FE on (979, 1568, 3) shards
В результате формируется граф знаний, приведенный в Приложении А.
```

Заключительный тест - это тест SPARQL-запроса select * where { ?s ?p ?o } к этому графу знаний:

```

curl -X POST http://localhost:6543/sa-1.0/sparql/a535ef20-2b7b-11ef-9f92-704d7b84fd9f/data-urlencode 'query=select * where { ?s ?p ?o }' -H "Accept: text/turtle" -H 'Content-Type: application/x-www-form-urlencoded'
{"results": {"bindings": [{"s": {"type": "uri", "value": "https://irnok.net/ontology/shores-2b7b-11ef-9f92-704d7b84fd9f"}, "p": {"type": "uri", "value": "https://irnok.net/ontology/shores-2b7b-11ef-9f92-704d7b84fd9f-0-mask"}, "o": {"type": "uri", "value": "https://irnok.net/ontology/shores-2b7b-11ef-9f92-704d7b84fd9f-1-segmentation-bbox"}, "p": {"type": "uri", "value": "http://www.w3.org/ns/rdf-syntax-ns#type"}, "o": {"type": "uri", "value": "https://irnok.net/ontology/shores-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation-bbox"}, "p": {"type": "uri", "value": "http://www.w3.org/ns/rdf-syntax-ns#type"}, "o": {"type": "uri", "value": "https://irnok.net/ontology/shores-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation-bbox"}, "p": {"type": "uri", "value": "http://www.w3.org/ns/rdf-syntax-ns#type"}, "o": {"type": "uri", "value": "https://irnok.net/ontology/shores-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation-crop-box"}, "p": {"type": "uri", "value": "http://www.w3.org/ns/rdf-syntax-ns#type"}, "o": {"type": "uri", "value": "https://irnok.net/ontology/shores-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation"}, .....}.....
```

В результате возвращается ответ в форме JSON-LD, содержащий все тройки графа знаний.

Исходный код сервера находится по адресу <https://github.com/eugeneai/shores-server>

ЗАКЛЮЧЕНИЕ

Выпускная квалификационная работа посвящена проекту развития научного направления Лаборатории инженерной геологии и геоэкологии в аспекте повышения уровня информатизации и обработки данных полевых исследований. В работе решены следующие задачи:

1. Проведена формализация предметной области инженерной геологии, относящейся к исследованиям экзогенных геологических процессов на берегах крупного внутреннего водоёма, представлена концептуальная модель в виде онтологии,
2. Разработана информационно-вычислительной инфраструктура поддержки прогнозирования состояния береговой зоны на основе геоинформационной системы и распределенной вычислительной среды,
3. Предложен вариант архитектуры вычислительной среды, реализованы некоторые её подсистемы с использованием современных средств автоматизации распознавания объектов на изображениях,
4. Для реализованных подсистем предложены модели данных, предназначенных для хранения информации в процессе её обработки,
5. Продемонстрирована работоспособность предложенной среды на простом примере прогнозирования.

Рассмотренные в диссертации вопросы преследуют целью формирование вычислительных ресурсов поддержки принятия решений по результатам мониторинга, оценки и прогноза опасных геологических процессов. Дальнейшее направление научных исследований и опытно-конструкторских работ имеет смысл продолжать по нескольким основным направлениям:

- Завершение реализации информационно-вычислительный среды,
- Наполнение информационных ресурсов архивными данными и данными современного мониторинга объектов исследования,
- Совершенствование методов прогнозирования состояния береговой зоны за счет реализации современного уровня информационного обеспечения,
- Разработка экспертной системы оценки результатов моделирования с целью формирования рекомендаций по использованию конкретных участков исследуемой береговой зоны.

Преимущества рассмотренного подхода – это а) отсутствие трудоемкого этапа подготовки данных для обучения НС, б) распознавание контура представляется (программируется) в виде правил, что дает возможность управлять процессом распознавания, в частности “сцеплять” объекты или контуры с разных изображений; в) SA, ввиду обученности на большом количестве изображений не привязана к свойствам конкретных изображений (размеру, цветовой гамме, повороту и т.д.), что

позволяет г) реализовывать (в перспективе) процедуры последовательного уточнения характеристик береговой линии, переходя к изображениям более высокого разрешения.

Разработанная платформа и модели обеспечат более эффективный способ оцифровки данных с изображений, результаты будут востребованы во многих проектных, изыскательских и научно-исследовательских организациях для проведения комплексных оценок, прогнозирования и принятия решений по управлению различными объектами береговой зоной для обеспечения их безопасного и рационального использования.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Thomas R. Gruber, Toward principles for the design of ontologies used for knowledge sharing?, International Journal of Human-Computer Studies, Vol. 43, No. 5–6, 1995, Pp. 907-928, ISSN 1071-5819, doi:10.1006/ijhc.1995.1081, <https://www.sciencedirect.com/science/article/pii/S1071581985710816> (дата доступа: 11.05.2024)
2. Dang K. B., Vu K. C., Nguyen H., Nguyen D. A., *et al.*. Application of deep learning models to detect coastlines and shorelines. // Journal of Environmental Management, 2022, No. 320, 115732.
3. Seale C., Redfern T., Chatfield P., Luo C., Dempsey K. Coastline detection in satellite imagery: A deep learning approach on new benchmark data // Remote Sensing of Environment, 2022, No. 278, 113044.
4. Aghdami-Nia M., Shah-Hosseini R., Rostami A., Homayouni S. Automatic coastline extraction through enhanced sea-land segmentation by modifying Standard U-Net // International Journal of Applied Earth Observation and Geoinformation, 2022, No. 109, 102785.
5. Сайт проекта OpenStreetMap – URL:<https://www.openstreetmap.org/> (дата обращения: 15.05.2024)
6. Геоинформационные системы: что это за технология и как работает – URL:<https://trends.rbc.ru/trends/industry/61f8fb399a7947618807cc41> <https://gis-lab.info/qa/saga-intro.html>
7. Официальный сайт редактора онтологий Protégé. – URL:<https://protege.stanford.edu/>
8. Официальный сайт проекта Linked Open Vocabularies (LOV). URL:<http://lov.okfn.org/dataset/lov/>
9. Макаров В. З. и др. Опыт создания геоинформационных систем на географическом факультете Саратовского университета // Известия Саратовского университета. Новая серия. Серия Науки о Земле. – 2008. – Т. 8. – №. 1. – С. 7-15.
10. Овчинников Г. И., Павлов С. Х., Тржцинский Ю. Б. Изменение геологической среды в зонах влияния Ангаро-Енисейских водохранилищ. Новосибирск: Наука, 1999. 254 с.
11. Овчинников Г. И. Динамика береговой зоны ангарских водохранилищ : автореф. дис. ... д-ра геогр. наук : 25.00.25. Иркутск, 2003. 50 с.
12. Рагозин А.Л., Бурова В.Н., Егоров А.Я. Способ локального экспресс-прогноза переработки берегов водохранилищ. – 1999.
13. Качугин Е. Г. Инженерно-геологические исследования и прогнозы переработки берегов водохранилищ. - В кн.: Рекомендации по изучению переработки берегов водохранилищ. - М.: Госгеолтехиздат, 1959, - С. 3-89.

14. Кондратьев Н.Е. Прогноз переформирования берегов водохранилищ под действием ветровой волны. - Труды Гос. гидролог. ин-та. - вып. 58(110), 1956, - С. 12-19.
15. Золотарев Г.С. Инженерно-геологическое изучение береговых склонов водохранилищ. - Труды 7-го Байкальского науч. координац. совещ. по изучению водохранилищ. -М., 1961.-т. 1, -С. 50-63.
16. Розовский Л. Б. О прогнозе переработки берегов водохранилищ по аналогии. - М.: "Наука 1961, - С. 45-57.
17. Епишин В.К., Экзарьян В.Н. Прогноз процесса формирования берегов водохранилищ. - М.: Энергия, 1979, - С. 95-111.
18. Гречищев Е К Метод расчета ширины зоны размыва берегов на примере Братского водохранилища Иркутск Кн изд-во» 1961 91 с.
19. Качугин Е. Г. Инженерно-геологические исследования и прогноз переработки берегов водохранилищ. М.: Госгеолтехиздат, 1959. 89 с.
20. Кондратьев Н. Е. Расчеты берегов переформирований на водохранилищах (практическое руководство). Л.: Гидрометеоиздат, 1960. 64 с.
21. Рагозин А. Л., Бурова В. Н. Метод прогнозной экспресс-оценки интенсивной переработки берегов водохранилищ С Гидротехническое строительство. 1993. № 10. С. 20—26.
22. Рекомендации по инженерным изысканиям для прогноза переработки берегов. М.: Стройиздат, 1986. 56 с.
23. Розовский Л. В., Зелинский И. П. Инженерно-геологические прогнозы и моделирование. Одесса, 1975. 115 с.
24. Иванов И. П., Тржцинский Ю. Б. Инженерная геодинамика // С-Пб.: Наука. – 2001. 411 с.
25. Козырева Е. А., Бабичева В. А., Мазаева О. А. Трансформация геологической среды в зоне влияния водохранилищ Ангарского каскада ГЭС //Известия Иркутского государственного университета. Серия: Науки о Земле. – 2018. – Т. 25. – С. 66-87.
26. Isha I. B., Adib M. R. M. Application of Geospatial Information System (GIS) using Digital Shoreline Analysis System (DSAS) in Determining Shoreline Changes //IOP Conference Series: Earth and Environmental Science. – 2020. – Т. 616. – №. 1. – С. 012029.
27. Matin N., Hasan G. M. J. A quantitative analysis of shoreline changes along the coast of Bangladesh using remote sensing and GIS techniques //Catena. – 2021. – Т. 201. – С. 105185.
28. Kabuth A. K., Kroon A., Pedersen J. B. T. Multidecadal shoreline changes in Denmark //Journal of Coastal Research. – 2014. – Т. 30. – №. 4. – С. 714-728.

29. Himmelstoss E. Аю, Thierler E. R., Zichichi J., & Ergul A. 2009 DSAS 4.0 Installation Instructions and User Guide. Updated for version 4.3 (only compatible with ArcGIS 10). In U.S. Geological Survey Open-File Report 2008-1278
30. Himmelstoss E. A., Henderson, R. E., Kratzmann, M. G., & Farris, A. S. Digital Shoreline Analysis System (DSAS) Version 5.0 User Guide: US Geological Survey Open-File Report 2018-1179 //Digital Shoreline Analysis System (DSAS) version 5.0 user guide: US Geological Survey Open-File Report 2018-1179. – 2018.
31. Геоинформационные системы: что это за технология и как работает – URL:<https://trends.rbc.ru/trends/industry/61f8fb399a7947618807cc41> (дата доступа: 15.05.2024).
32. UMBEL Ontology<https://lov.linkeddata.es/dataset/lov/vocabs/umbel> (дата доступа: 15.05.2024)
33. Сделай SAM: Segment Anything Model в задачах компьютерного зрения. – <https://habr.com/ru/companies/sberdevices/articles/757606/> (дата доступа: 15.05.2024)

ПРИЛОЖЕНИЕ А Граф знаний, полученный анализом масок изображения, формат turtle

```
1 @prefix foaf: <http://xmlns.com/foaf/0.1/> .  
2 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .  
3 @prefix schema: <https://schema.org/> .  
4 @prefix sd: <https://irnok.net/ontology/shores/a/1.0/> .  
5 @prefix shape: <https://irnok.net/ontology/shores/t/1.0/> .  
6 @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .  
7  
8 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f a foaf:Image ;  
9     rdfs:label "Bykovo.jpg"@en ;  
10    shape:mask sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-mask,  
11        sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-mask,  
12        sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-mask,  
13        sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-mask,  
14        sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-mask,  
15        sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-mask,  
16        sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-mask,  
17        sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-mask ;  
18    shape:shape sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-shape .  
19  
20 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-mask a shape:Mask ;  
21     shape:area 674868 ;  
22     shape:predicted_iou 1.004482e+00 ;  
23     shape:segmentation sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-segmentation ;  
24     shape:stability_score 9.672641e-01 ;  
25     schema:sku 0 .  
26  
27 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-segmentation a shape:Segmentation ;  
28     shape:bbox sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-segmentation-bbox ;  
29     shape:crop-box sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-segmentation-crop-box ;  
30     shape:point-coords  
     ↳ sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-segmentation-point-coords ;
```

```

31     shape:shape sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-shape .
32
33 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-segmentation-bbox a shape:Bbox ;
34     shape:height 945 ;
35     shape:left 625 ;
36     shape:top 0 ;
37     shape:width 942 .
38
39 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-segmentation-crop-box a shape:Crop-box ;
40     shape:height 979 ;
41     shape:left 0 ;
42     shape:top 0 ;
43     shape:width 1568 .
44
45 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-0-segmentation-point-coords a shape:Point-coords
46     ↳ ;
47     shape:x 1.1515e+03 ;
48     shape:y 6.271719e+02 .
49
50 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-mask a shape:Mask ;
51     shape:area 528490 ;
52     shape:predicted_iou 9.925082e-01 ;
53     shape:segmentation sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-segmentation ;
54     shape:stability_score 9.562669e-01 ;
55     schema:sku 1 .
56
57 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-segmentation a shape:Segmentation ;
58     shape:bbox sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-segmentation-bbox ;
59     shape:crop-box sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-segmentation-crop-box ;
60     shape:point-coords
61     ↳ sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-segmentation-point-coords ;
62     shape:shape sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-shape .

```

```

63     shape:height 966 ;
64     shape:left 0 ;
65     shape:top 5 ;
66     shape:width 745 .
67
68 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-segmentation-crop-box a shape:Crop-box ;
69     shape:height 979 ;
70     shape:left 0 ;
71     shape:top 0 ;
72     shape:width 1568 .
73
74 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-1-segmentation-point-coords a shape:Point-coords
75     → ;
76     shape:x 2.695e+02 ;
77     shape:y 7.648438e+01 .
78
79 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-mask a shape:Mask ;
80     shape:area 21521 ;
81     shape:predicted_iou 9.818562e-01 ;
82     shape:segmentation sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-segmentation ;
83     shape:stability_score 9.713498e-01 ;
84     schema:sku 2 .
85
86 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-segmentation a shape:Segmentation ;
87     shape:bbox sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-segmentation-bbox ;
88     shape:crop-box sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-segmentation-crop-box ;
89     shape:point-coords
90         → sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-segmentation-point-coords ;
91     shape:shape sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-shape .
92
93 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-segmentation-bbox a shape:Bbox ;
94     shape:height 64 ;
95     shape:left 0 ;
96     shape:top 4 ;

```

```

95     shape:width 342 .
96
97 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-segmentation-crop-box a shape:Crop-box ;
98     shape:height 979 ;
99     shape:left 0 ;
100    shape:top 0 ;
101    shape:width 1568 .

102
103 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-2-segmentation-point-coords a shape:Point-coords
104     ↳ ;
105     shape:x 1.225e+02 ;
106     shape:y 1.529688e+01 .

107 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-mask a shape:Mask ;
108     shape:area 35433 ;
109     shape:predicted_iou 9.700564e-01 ;
110     shape:segmentation sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation ;
111     shape:stability_score 9.679427e-01 ;
112     schema:sku 3 .

113
114 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation a shape:Segmentation ;
115     shape:bbox sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation-bbox ;
116     shape:crop-box sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation-crop-box ;
117     shape:point-coords
118     ↳ sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation-point-coords ;
119
120 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation-bbox a shape:Bbox ;
121     shape:height 133 ;
122     shape:left 1296 ;
123     shape:top 807 ;
124     shape:width 268 .

125
126 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation-crop-box a shape:Crop-box ;

```

```

127     shape:height 979 ;
128     shape:left 0 ;
129     shape:top 0 ;
130     shape:width 1568 .
131
132 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-3-segmentation-point-coords a shape:Point-coords
133   → ;
134   shape:x 1.4945e+03 ;
135   shape:y 8.719219e+02 .
136
137 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-mask a shape:Mask ;
138   shape:area 2009 ;
139   shape:predicted_iou 9.498836e-01 ;
140   shape:segmentation sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation ;
141   shape:stability_score 9.720588e-01 ;
142   schema:sku 4 .
143
144 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation a shape:Segmentation ;
145   shape:bbox sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation-bbox ;
146   shape:crop-box sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation-crop-box ;
147   shape:point-coords
148     → sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation-point-coords ;
149   shape:shape sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-shape .
150
151 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation-bbox a shape:Bbox ;
152   shape:height 51 ;
153   shape:left 1501 ;
154   shape:top 95 ;
155   shape:width 49 .
156
157 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation-crop-box a shape:Crop-box ;
158   shape:height 979 ;
159   shape:left 0 ;
160   shape:top 0 ;

```

```

159     shape:width 1568 .
160
161 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-4-segmentation-point-coords a shape:Point-coords
162     ↳ ;
163     shape:x 1.5435e+03 ;
164     shape:y 1.070781e+02 .
165
166 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-mask a shape:Mask ;
167     shape:area 4866 ;
168     shape:predicted_iou 9.46198e-01 ;
169     shape:segmentation sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-segmentation ;
170     shape:stability_score 9.688007e-01 ;
171     schema:sku 5 .
172
173 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-segmentation a shape:Segmentation ;
174     shape:bbox sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-segmentation-bbox ;
175     shape:crop-box sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-segmentation-crop-box ;
176     shape:point-coords
177         ↳ sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-segmentation-point-coords ;
178     shape:shape sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-shape .
179
180 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-segmentation-bbox a shape:Bbox ;
181     shape:height 82 ;
182     shape:left 1487 ;
183     shape:top 6 ;
184     shape:width 80 .
185
186 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-segmentation-crop-box a shape:Crop-box ;
187     shape:height 979 ;
188     shape:left 0 ;
189     shape:top 0 ;
190     shape:width 1568 .

```

```

190 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-5-segmentation-point-coords a shape:Point-coords
191   ↳ ;
192   shape:x 1.5435e+03 ;
193   shape:y 7.648438e+01 .
194
195 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-mask a shape:Mask ;
196   shape:area 3749 ;
197   shape:predicted_iou 9.353169e-01 ;
198   shape:segmentation sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-segmentation ;
199   shape:stability_score 9.622642e-01 ;
200   schema:sku 6 .
201
202 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-segmentation a shape:Segmentation ;
203   shape:bbox sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-segmentation-bbox ;
204   shape:crop-box sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-segmentation-crop-box ;
205   shape:point-coords
206     ↳ sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-segmentation-point-coords ;
207   shape:shape sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-shape .
208
209 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-segmentation-bbox a shape:Bbox ;
210   shape:height 98 ;
211   shape:left 608 ;
212   shape:top 234 ;
213   shape:width 48 .
214
215 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-segmentation-crop-box a shape:Crop-box ;
216   shape:height 979 ;
217   shape:left 0 ;
218   shape:top 0 ;
219   shape:width 1568 .
220
221 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-6-segmentation-point-coords a shape:Point-coords
222   ↳ ;
223   shape:x 6.125e+02 ;

```

```

221     shape:y 2.600469e+02 .
222
223 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-mask a shape:Mask ;
224     shape:area 936 ;
225     shape:predicted_iou 9.145088e-01 ;
226     shape:segmentation sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation ;
227     shape:stability_score 9.767932e-01 ;
228     schema:sku 7 .

229
230 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation a shape:Segmentation ;
231     shape:bbox sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation-bbox ;
232     shape:crop-box sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation-crop-box ;
233     shape:point-coords
234         → sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation-point-coords ;
235
236 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation-bbox a shape:Bbox ;
237     shape:height 48 ;
238     shape:left 1296 ;
239     shape:top 863 ;
240     shape:width 29 .

241
242 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation-crop-box a shape:Crop-box ;
243     shape:height 979 ;
244     shape:left 0 ;
245     shape:top 0 ;
246     shape:width 1568 .

247
248 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-7-segmentation-point-coords a shape:Point-coords
249         → ;
250     shape:x 1.2985e+03 ;
251     shape:y 8.719219e+02 .

252 sd:a535ef20-2b7b-11ef-9f92-704d7b84fd9f-shape a shape:Shape ;

```

```
253     shape:colors 3 ;
254     shape:cols 1568 ;
255     shape:rows 979 .
```