ESPRESSO Use Cases

In response to this call to the Solid Practitioners community from Jeff Zucker: https://github.com/solid-contrib/practitioners/discussions/10 we describe below five use cases for the ESPRESSO experimental search framework.

These scenarios, like everything else about the ESPRESSO project, are a work in progress. We welcome your comments, questions, suggestions and feedback.

You should join us at the DESERE Workshop in Singapore on 13 May 2024!

More info: https://arxiv.org/abs/2403.07732

Register: https://www2024.thewebconf.org/attending/registration/

You should also come to our workshop in London on 13-14 June 2024! Watch this space for details.

ESPRESSO is a joint project between the University of Southampton and Birkbeck, University of London, funded by EPSRC. The NUS Extreme Search Centre (NExT++) and Dataswyft are project partners.

ESPRESSO is researching large-scale data search across distributed Solid servers by developing and evaluating decentralised algorithms, meta-information data structures, and indexing techniques. The aim is to support both keyword-based search and SPARQL querying. To achieve this, we are factoring information about owners' data access and caching restrictions into our algorithm, data structure, and index design. We use generic and domain-specific scenarios to drive development and evaluation.The generic scenarios include (i) exploratory keyword-based search (both top-k and exhaustive) across a large scale of pods to discover information or to build communities; (ii) a large number of users posing distributed SPARQL queries over a large scale of pods; and (iii) community-based keyword-based search and querying, where access to query endpoints and data is specified at the level of community membership. Domain-specific scenarios from health and wellbeing domain are also used.

The architecture that we use for experimentation includes the following components:

- The ESPRESSO Indexing app, which indexes pod data, requires read access to every resource which the pod owner wishes to make available for search. The indexer requires write access to the pod, as the privacy principle requires the index to stay on the pod. Minimal, privacy-preserving information about the indexed data is stored by ESPRESSO on ESPRESSO's own pod.
- The ESPRESSO Search app searches all the indexed resources to which the search party's WebID has read access. The search app needs read access to each pod's index file.
- The ESPRESSO overlay network (currently based on GaianDB) routes the queries efficiently over Solid servers, and will show each search party a different view, including different ranking, for the search results depending on the access control granted to the search party's WebID.

We are experimenting with different deployment settings, where each Solid server can host numbers of pods ranging from one pod to thousands, and different distributions of pods among Solid servers will be trialled. This allows us to explore the challenges of search in settings where a large number of fine-grained access control and caching policies are managed in a single Solid server on the one hand, and where query propagation on a large scale needs to be conducted, taking access control and caching restrictions of a large number of SOLID peers into account, on the other. We also consider a range of intermediate settings in between these two extremes.

The following are some non-exhaustive examples of health and well-being scenarios that we consider:

Scenario 1: A Clinical Trial

Alice, a medical researcher working at the Medical Research Institute in the UK, wants to search for possible participants to take part in an approved first stage clinical trial for a new diabetes treatment. Alice wants to select participants based on the information in their medical records.

The Task: Alice, a medical researcher working at the Medical Research Institute in the UK wants to search for possible participants to take part in an approved first stage clinical trial for a new diabetes treatment. Alice wants to select participants based on the information in their medical records.

Centralized Conditions

The Problem: Currently, confidential patient data is held in data silos by data controllers who ensure its protection. In this scenario, the data controllers are the National Health System (NHS) hospital trusts that the patients were admitted to. However, NHS data silos hamper data discoverability and usage even by parties such as Alice who are authorized for its access and use. Patients can use an NHS app to download a partial copy of their medical records, but cannot make the downloaded copy searchable or discoverable mwithout loss of confidentiality.

- environment: NHS trust servers
- host: NHS
- data storage: NHS trust silos
- data type: confidential patient records
- data owner(s): NHS trusts
- permissions from/to: NHS Trusts via patients/Alice via MRI
- grantee identified by: NHS Trusts' auth process
- grantee: Alice via MRI
- grantee accesses: confidential patient records
- query format: keyword
- query format: structured

Decentralized Conditions

The Solution: Patients can download a partial copy of their medical records from an NHS app. They can store this copy of their records on their personal pod hosted by the provider of their choice, or on a patient pod hosted on servers run by the NHS and/or a consortium of medical research institutes. These patient pods are created to make it easy for people to share their data with trusted institutions for research purposes, and to be contacted about opportunities to volunteer for clinical trials. As pod owners, they can choose which organizations' WebIDs have access to which resources in their patient pods.

Alice uses ESPRESSO to search the patient pods for the mentions of diabetes using MRIWebID on the institutional servers, and obtains a list of people who have diabetes mentioned in their medical history and whom she can contact. Alice invites the patients who meet the criteria to volunteer for the trial.

Additional considerations: If there is a standardized format for medical data (like Fast Health Interoperability Resources (FHIR) data integration schema), a structured query could be tailored to that format instead of using a keyword search.

Search Scope: The scope is limited to the institutional servers where the patient pods are held. Geographically, the search is limited to the UK (assuming all the patient Solid servers are located in the UK). This search is organizational (the search party uses MRIWebID which belongs to an organization), and is a keyword search.

- search party: organization (MRI)
- search type: keyword
- server scope: patient Solid servers only
- pod owners: those with MRIWebID in their acls
- acls: MRIWebID
- filetypes: patient records are held in several different coding systems, plus various kinds of unstructured/free text data, but the copy patients download is formatted by the app
- other, eg location: N/A (assume search is geographically limited to the UK)

Scenario 2. Training Program

The Task: Bob, a fitness coach wants to introduce a new training program to their group, but this contains exercises that might not be suitable for people with carpal tunnel syndrome. Bob wants to know if some people in the group have carpal tunnel syndrome so he can alter the program as needed.

Centralized Conditions

The Problem: Currently, it is impossible in any classical search system for users to give partial access for search over their sensitive data, beyond research prototypes. Bob might have to resort to asking the people in the group directly, for example by asking them to fill out a questionnaire or a form.

- environment: centralized Web
- host: questionnaire software vendor e.g. Google Forms
- data storage: vendor's server
- data type: special category data
- data owner(s): questionnaire software vendor
- permissions from/to: Bob / questionnaire software vendor
- permissions from/to: group member / questionnaire software vendor
- permissions from/to: group member / Bob
- grantee identified by: questionnaire software vendor's auth protocol
- grantee: questionnaire software vendor
- grantee: Bob
- grantee accesses: special category data
- query format: structured

Decentralized Conditions

The Solution: Bob uses ESPRESSO to search for the mentions of carpal tunnel syndrome using his WebID in the pods belonging to the WebIDs from the list.

Additional considerations: There could be a standardized way (through an app) that, based on a user's medical data, produces a summary in a known format that the user can then store in their pod. This way, the search query can again be a structured query tailored to that format.

New Training Program Scenario: The scope is limited to a list of specific pods associated with the group members' WebIDs, and we are only interested in files discoverable by the search party's (Bob's) WebID. This is a personal search (Bob is a person), done by keyword.

- search party: personal (Bob)
- search type: keyword
- search type: structured (if Bob has an app)
- server scope: N/A
- pod owners: list of group members' WebIDs
- acls: readable by Bob's WebID

- filetypes: Bob may or may not know the target file format in advance
- other, eg location: N/A

Scenario 3: New Fitness Group

The Task: Carol, another fitness coach, wants to compile a training program for a new group. They need to gauge the average fitness level of the group to make sure the program is suitable for all the attendees by accessing the information generated by their wearable fitness monitors.

Centralized Conditions

The Problem: In commonly used architectures this type of information is stored centrally on the app servers and is inaccessible to search engines. The user would have to export the relevant information (if possible) and share it with Carol manually.

- environment: vendors' servers
- host: wearable device vendors
- host: wearable device users' local systems and/or their storage providers
- data storage: vendors' central servers
- data type: health/wellness data from consumer wearable devices
- data owner(s): wearable device vendors
- data owner(s): wearable device users
- permissions from/to: users/wearable device vendors
- permissions from/to: users/Carol
- grantee identified by: user login
- grantee: wearable device vendors
- grantee: Carol
- grantee accesses: health/wellness data from consumer wearable devices
- query format: keyword
- query format: structured

Decentralized Conditions

The Solution: fortunately for Carol, someone has developed that app that Bob wanted, for sharing data like this amongst fitness professionals and between enthusiasts in a standard format. Carol can now use ESPRESSO's search app to send structured queries over this data and find out the values of various group fitness indicators, such as the average, maximum, or minimum number of steps taken daily by the group members.

Additional considerations: We assume the data is stored in a known format because of Carol's fitness data sharing app, whose developers will have done the necessary schema mapping. The queries can be further tailored to prevent erroneous exclusion of people with different conditions, for example, potential attendees with mobility impairments or other conditions that limit their daily step count, but who are otherwise fit enough to attend.

New Fitness Group Scenario: The scope is limited to specific pods associated with Carol's group members' WebIDs; and by metadata (the search party is interested only in wearable fitness monitor data). This is a personal and structured (we are expecting to get numerical values) search.

- search party: personal (Carol)
- search type: structured
- server scope: N/A
- pod owners: group members' WebIDs
- acls: readable by Carol's WebID
- filetypes: wearable fitness monitor data filetypes
- other, eg location: N/A

Scenario 4: Contact Tracing App

The Task: COVIDTrackr, a contact tracing app, allows the user to 'check in' physically at a location, for example, by scanning a QR code or selecting the location from a pre-populated list and notifies a user if they were recently at the same location as a person who was shortly afterwards diagnosed with COVID-19. COVIDTrackr does that without revealing who the contacted person was and without centrally storing the geolocation information of its users.

Centralized Conditions

The Problem: Usually, such apps have to store and aggregate their data centrally. This leads to concerns about individual data privacy, and potential misuse of the data. Also, the infrastructure for the app must include substantial space to store the data.

- environment: app vendors' servers
- host: app vendors or app vendors' storage providers
- data storage: app vendors' servers or those of their storage providers
- data type: health/wellness data from mobile phone sensors
- data owner(s): app vendors
- permissions from/to: app user/app vendor
- grantee identified by: themselves (app), app's auth process
- grantee: COVIDTrackr, COVIDTrackr user
- grantee accesses: health/wellness data from mobile phone sensors
- query format: keyword
- query format: structured

Decentralized Conditions

The Solution: When Dan, a user of COVIDTrackr, notifies COVIDTrackr about a positive test, COVIDTrackr uses ESPRESSO search to execute a structured query over all its data in all of the users' pods to determine who was at the same location at the same time as Dan. It then individually notifies all the users in the search result about potential contact with the time and place. The notified user does not receive any information that they do not already have, except that they had a potential contact with someone who recently tested positive. There is no centralized server or ability to mine a list of check-ins for an individual; all personal data is within the individual's personal control.

Additional considerations: After being notified by Dan, COVIDTrackr can repeat the search periodically, to reduce the risk of missing any personal pods that go offline for some time.

New Contact Tracing App Scenario: We limit the scope to specific pods (belonging to the users of the app) and by metadata (the app data). This is an app (the search party is the app) and structured search.

- search party: app
- search type: structured
- server scope: N/A
- pod owners: app users
- acls: N/A presumably because all app data is readable by the app
- filetypes: known app file types
- other, e.g. location: location is the core data

Scenario 5: Activity Level Study

The Task: Eve, a researcher working at the Medical Research Institute in the UK, wants to do a pre-clinical study on the relationship between activity levels and post-operative recovery time. Eve wants to select participants whose medical records show that they have undergone surgery. Eve wants to integrate this data with the step count information from an individual's wearable fitness monitor.

Centralized Conditions

The Problem: Confidential patient data is stored in silos that even authorized parties, such as Eve, often have difficulty accessing. Patients can download a partial copy of their medical records using an NHS app, but cannot make the copy searchable or discoverable without loss of control or confidentiality.

Combining patient records with wellness data from commercial fitness monitors would add real-time insight about the study volunteers' actual everyday behaviour, but it is typically stored on the device vendors' servers and is not searchable or discoverable by classic centralized search engines – nor should it be, as special category data. Users should be able to obtain a copy of their data under GDPR, but to make the best use of it they need to be able to store it securely and share it without losing control of it.

This scenario has the same constraints as Scenario 1 with respect to the availability of medical data. It also has the added complication of identifying available individual wellness information (e.g. step count) and integrating it with the medical data.

Decentralized Conditions

The Solution: Each patient in this scenario has downloaded a partial copy of their patient records via an NHS app. Responding to an initiative that enables patients to make their medical records available to researchers, and be contacted for volunteering opportunities, the NHS (and/or a consortium of medical research institutions) hosts dedicated Solid servers where each patient volunteer can store the copy of their records in their own patient pod.

Some patients have also downloaded their fitness tracker data and are keeping that on a Solid pod as well. This could be the patient pod, or their personal pod hosted by the provider of their choice. Both pods would be identified by the same WebID.

Eve's institutional MRIWebID lets her use ESPRESSO to search the patient pods for potential volunteers. These patient pods are all held on institutional servers, so Eve doesn't have to do a global search. After she has her list WebIDs of patients who meet the study criteria, she then has a finite list of patient and/or personal pods to search for fitness tracker data to which the MRIWebID has read access.

In some cases, patients will not want their medical records to be made directly searchable, but will be willing to make them discoverable in summary form. This may be all Eve needs to know at the recruitment stage.

Ideally, the fitness tracker app would have an option to share the data on Solid pods for medical research purposes, so the user does not have to acquire and store it manually. Once Solid pods have reached this level of mainstream adoption, fitness data from various vendors should eventually converge on a format that is easier to query.

Now all the data is in place, ready for Eve to do her ESPRESSO search. First, she does a keyword search over the patient pods, looking for patients who have had surgery. Once she has narrowed down a set of WebIDs of patients matching the study criteria, she first searches the patient pods for step counts (since patients can keep their fitness data in their patient pods) and then searches the personal pods for step counts (where the pod owners in Eve's list may be keeping their fitness data, if they aren't keeping it in their patient pods).

First search:

- search party: organization (MRI)
- search type: keyword
- server scope: patient Solid servers only
- pod owners: those with MRIWebID in their acls

- acls: MRIWebID
- filetypes: whatever type users of the NHS App receive their downloaded copy in
- other, eg location: N/A (assume search is geographically limited to the UK)

Second search:

- search party: organization (MRI)
- search type: structured
- server scope: patient and/or personal pods owned by the MRIWebIDs returned by the first search
- pod owners: those with MRIWebID in their acls
- acls: MRIWebID
- filetypes: the types known to be used by fitness trackers
- other, eg location: N/A