

The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

An openEHR 2025 fellowship project

Servei Català de la Salut

April 2026

Direction or coordination:

Modelatge del coneixement clínic / Eix Estratègic / Pla Director de Sistemes d'Informació / Àrea de Sistemes d'Informació

Author:

Martin Andreas Koch, PhD

Some rights reserved:

© 2026, Government of Catalonia. Catalan Health Service.



The contents of this work are subject to a Attribution-NonCommercial-NoDerivatives 4.0 International license.

The license can be consulted on the <https://creativecommons.org/licenses/by-nc-nd/4.0/deed.ca> website.

Publisher:

Catalan Health Service

1st edition:

Barcelona, April 2026

Subject descriptors::

openEHR, fellowship, project, archetypes

Accessible document (2026, version 1)

Table of Content

1. Project Context.....	10
1.1. Reference.....	11
2. Project approach	12
3. Initial User Questionnaire	12
3.1. Results of the Initial User Questionnaire	13
Section I: Getting to Know the Professional	13
Section II: Drafting and Compiling a List of Data Elements for an openEHR Template	17
Section III: Searching Archetypes.....	27
Optional Final Section: Suggestions and Feedback.....	35
3.2. Key Findings of the Questionnaire	37
4. Value Proposition	38
4.1. The "Customer" side.....	39
4.2. The "Value" side	39
4.3. Summary and formulation of the value proposal	40
5. Project planning.....	41
5.1. Research, Development, and Implementation Strategies.....	41
Research.....	41
Development	41
Implementation.....	41
5.2. Timing by quarters.....	43
Quarter 1 (Months 1-3).....	43
Quarter 2 (Months 4-6).....	43
Quarter 3 (Months 7-9).....	44
Quarter 4 (Months 10-12)	44
5.3. Detailed timing	45
6. Scope	46
6.1. In Scope	46
6.2. Out of Scope.....	46
7. Ethical Considerations.....	47
7.1. Questionnaire(s).....	47
7.2. Web-Based Tool.....	47
8. ADL extraction.....	48
8.1. Introduction.....	48

8.2.	ADL extraction GUI.....	49
8.3.	Data Source.....	52
	GitHub repositories.....	52
	CKM repositories.....	53
8.4.	Data Elements.....	55
	Definition of Data to be extracted	55
8.5.	Parsing ADL Files.....	57
	ANTLR4 approach.....	57
	Regex approach.....	58
	Incoherent archetype quality encountered during parsing.....	70
8.6.	Data Analysis.....	74
	Relationships: Parent-child, Inclusion and Exclusion	74
	Archetype Classes	75
	Lifecycle States	76
	Date of Authorship.....	78
	Keywords	80
8.7.	Recommendation.....	82
	When to Recommend.....	83
	How to Recommend.....	83
	Adding content for content-based filtering.....	84
	Content-Based Filtering by Semantic Linking of Archetypes.....	87
	Considerations for the Future	93
	References.....	93
8.8.	Project Publication and Source Code.....	93
	Source Code	93
	Executable Release	93
8.9.	Automation	94
	Requirements.....	94
	Automation YAML File.....	95
	Additional Setup in GitHub	96
8.10.	Detailed description of use.....	97
9.	Visualization Tool – The Archetype Companion	97
9.1.	Introduction.....	97
	General Description.....	97
	Key Features.....	98
	Other Requirements	101
9.2.	User Journey Mapping.....	102

User Persona: Clinical Data Modeller	102
User Journey	103
Summary of Design Requirements for Different Views/Pages/Windows	107
9.3. GUI	110
Wireframe/Early draft	110
First Iteration of the GUI	116
Second Iteration of the GUI	117
Colour themes	125
Third Iteration of the GUI	125
9.4. Detailed description of function	135
9.5. How elements and archetypes are connected in the Archetype Companion	135
Step 1: Define data elements	135
Step 2: Create an archetype collection	135
Step 3: Link elements to archetypes	135
Step 4: Use the links in the planner (structure)	135
Compact summary	136
9.6. Visualization Language	136
Archetype Connection	136
Archetype Structure	138
Similar Archetypes	138
Collection	140
Project	141
Template planner	141
9.7. Local storage	142
Why This Matters	142
Google Chrome	143
Mozilla Firefox	143
Microsoft Edge	143
Safari	144
10. Added value – The openEHR class glossary	145
10.1. Problem Statement	145
10.2. Existing Solutions	145
10.3. Proposed Solution	145
10.4. Implementation	145
10.5. Features	146
10.6. Detailed description of use	147
11. First user test (Soft launch)	147

11.1. Summary of Volunteer Responses: Archetype Companion User Testing.....	147
Methodology.....	147
Cohort	148
11.2. Final survey	149
Overall Sentiment and Key Strengths.....	149
Main Pain Points and Issues	150
11.3. Prioritized Improvements and Importance.....	152
12. Changes implemented before final release.....	153
12.1. P1 - Fix Template Planner text fields (selection & editing) – SOLVED	153
12.2. P1 - Fix Firefox dropdown rendering & truncation – SOLVED	153
12.3. P1 - Ensure all tutorial videos load reliably (esp. 6 & 9) - SOLVED	154
12.4. P2 - Auto-load archetype structure in Template Planner - SOLVED	154
12.5. P2 - Enhance graph navigation with “up one level” control - SOLVED	155
12.6. P2 - Auto-generate archetype IDs from concept name & class - SOLVED	156
12.7. P3 - Add mind map view for archetypes/templates - SOLVED	156
13. Final publication	157
14. Conclusions.....	158
14.1. Reflections on the Fellowship Programme	158
14.2. Project Process and Enabling Factors	158
14.3. Technical Learning and Perspective	158
14.4. Reflections on Community Infrastructure and Publishing	159
14.5. Contribution of the Project and Future Directions.....	159
14.6. Overall Closing Remarks	160
15. ANNEX I - ADL Extraction Tool – User Manual	161
15.1. Overview	161
15.2. Requirements and installation.....	161
Software Requirements	161
Installation	161
15.3. Usage	161
A. Running the Tool Manually (GUI Mode)	161
B. Running in Headless Mode (Automation)	162
15.4. Automation with GitHub Actions.....	163
Example Workflow (.github/workflows/update-dataset.yml):	163
15.5. Output.....	163
15.6. Troubleshooting.....	164
15.7. Support & Contributions.....	164
16. Annex II – Archetype Designer – User Manual	165

16.1.	Licence.....	165
16.2.	Welcome.....	165
16.3.	Video Tutorials.....	166
16.4.	Example Projects.....	166
16.5.	A brief overview of the interface.....	166
16.6.	About Page and Manual	169
16.7.	Managing a Project.....	170
	Create a Project	171
	Save a Project	172
	Load a Project.....	172
	Delete a Project	173
	Create a Report.....	173
	Other Project Options.....	175
16.8.	Navigate and Search for Archetypes	176
	Navigating the Archetype List	176
	The Search View	178
	Guide: Search Strategies in openEHR Archetypes for Clinical Modelling (Beginner friendly)	181
	Archetype information.....	185
	History.....	187
16.9.	Managing a Collection	189
	Add an Existing Archetype	189
	Remove Existing Archetype.....	190
	Hide or Show Collection Archetypes	191
	Create a "New" Archetype	192
	Edit a "New" Archetype.....	193
	Delete a "New" Archetype.....	198
16.10.	Managing Checklists	199
	Where to find the checklist	199
	Creating a Checklist in the Editor	200
	Assigning Archetypes to Data Elements.....	204
	Flagging Data Elements	205
	Hide/Show "approved" checklist items.....	205
	Save Checklist to Excel.....	206

Open Checklist from Excel File	206
Delete a Checklist from Project	207
16.11. Managing a Template-like Structure (Template Planner)	207
Where to Find the Template Planner Editor.....	208
Selecting an Archetype	208
Add archetype elements automatically	209
Adding a Branch	209
Cloning a Branch	210
Deleting a Node or a Branch.....	210
Collapsing/Expanding Branches	211
Moving Nodes and Branches Around	211
Switching Between Archetype and Data Element	212
Commenting Nodes	214
Assigning "atcode" Item to Archetype or Element.....	215
Compatibility Warnings	216
Delete All Template Planner Data.....	217
Template Planner and Visualisation	218
Template Planner Data in the Report	219
16.12. Visualisation.....	219
Where to Find the Visualisation Window.....	220
Navigating the Visualisation	220
The Different Views.....	222
Export Visualisation View	226
16.13. Glossary	227
Where to Find the Glossary	227
How to Search the Glossary	228
Close the Glossary	228
16.14. Data Storage.....	228
Local Storage.....	228
Archetype Database.....	230
16.15. Running Archetype Companion Locally.....	231
Updating the Archetype Data Dictionary.....	231
17. Annex III – Reflective Journals	233

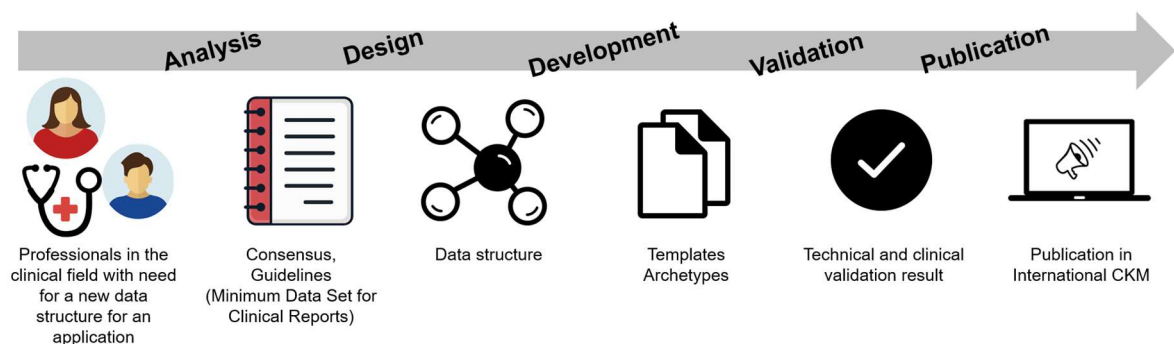
17.1. Reflective Journal 1	233
On the Fellowship.....	233
On My Project	233
17.2. Reflective Journal 2	236
Month 6... Already?	236
Month 5 fellowship session.....	236
Continuing Documentation	237
AI, or Not AI: That Is the Question.....	237
17.3. Reflective Journal 3	239
Continuing Work on the Tutorials	239
Presenting the Archetype Companion to the world.....	240
First user test concluded! The results are in!	241
Fellowship session Month 7 - The Standards Landscape	241
Fellowship session Month 8 - Real World Cases	242
17.4. Reflective Journal 4	243
Month 10 session - Patients, Providers, Payers, and Policy Makers.....	243
Month 11 session - Patients, Providers, Payers, and Policy Makers.....	243

1. Project Context

The project described here is meant to fit into the clinical modelling methodology. In general, this methodology consists of several steps. It begins with the **identification of a need** for a clinical model, based on a real clinical necessity. A multidisciplinary team conducts an **analysis** to determine all the necessary data elements. The team, drawing on professional experience and existing guidelines (e.g., the Minimum Data Set for Clinical Reports), should achieve consensus.

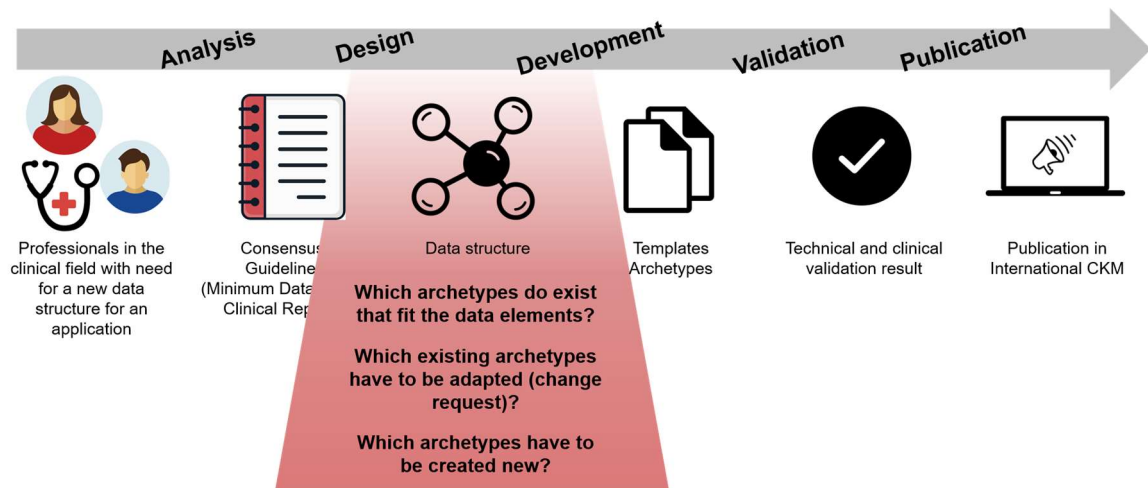
Once consensus is reached, the data structure is defined in the **design** phase. Here, the structure is specified, determining the occurrences and location of every element. This is guided not only by clinical necessity but also by technological frameworks. During the **development** phase, this structure is implemented into archetypes and templates using openEHR standards in a special development environment, such as the Archetype Designer (<https://tools.openehr.org/designer/>).

When the templates are ready for publication, a **validation** process is followed to ensure correct function from both clinical and technical perspectives. If validation is successful, the archetypes and templates are **published** in the international Clinical Knowledge Manager (CKM).



The openEHR clinical modelling workflow (based on Moner D. et al.)

Although real-life development is not always as linear as the schema suggests, there is a critical moment: when the data elements for the data structure are defined, **professionals must search for archetypes** that might contain these data elements. This is important, as the philosophy of openEHR is based on the reuse of assets and worldwide consensus on reusable archetypes.



Main questions arising in the design phase of the workflow.

At this stage, the professional must determine:

- Which archetypes already exist that fit the data elements?
- Which existing archetypes need to be adapted (change request)?
- Which archetypes need to be created anew?

The proposed project aims to assist professionals in this critical step, given the complexity of the archetype ecosystem.

1.1. Reference

Moner D, Maldonado JA, Robles M. Archetype modelling methodology. J Biomed Inform. 2018 Mar;79:71-81. doi: 10.1016/j.jbi.2018.02.003

2. Project approach

As this project is largely a software development effort, the following approach will be taken.

A **project plan** is created at the beginning to define scope, limitations, and deadlines.

To capture user needs, an **initial user survey** is conducted.

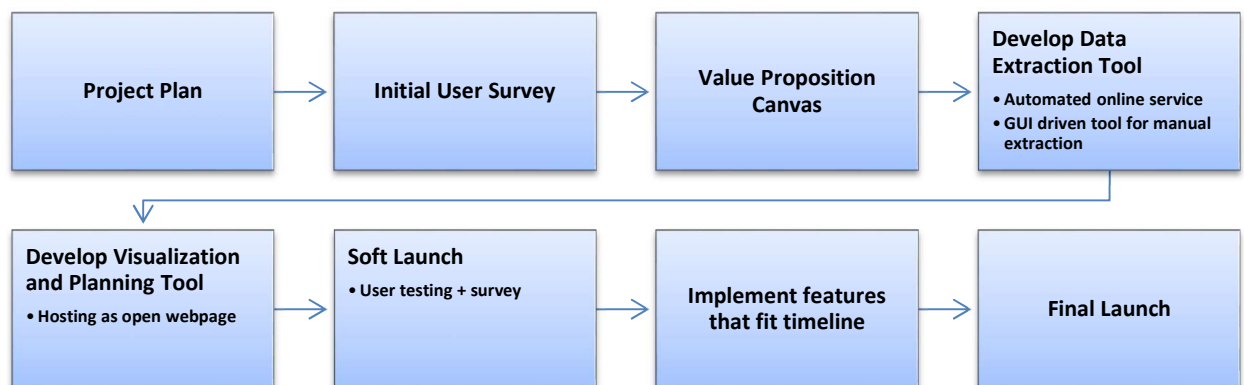
Based on the results, a **value proposition** is defined to address these needs.

Then, **software development** takes place.

After completion of the prototype, a **soft launch with a limited user group** is carried out to assess usefulness and identify required changes before the final launch.

The findings from this user test are categorized into **changes to implement now or later**.

Finally, the **software is launched to the public**.



3. Initial User Questionnaire

To capture information of the intended user, a questionnaire was designed to gather insights from professionals involved in clinical data modelling using openEHR. It covers participants' backgrounds, their approaches to drafting and compiling data elements, and their experiences with searching, planning, and mapping archetypes.

The survey also explores the tools and methods used for creating and validating openEHR templates, as well as challenges faced during the modelling process. We hope that the collected feedback will help guide the development of tools and processes that better support the needs of the openEHR community.

Specifically it was intended to confirm the necessity of the proposed project and to ascertain the most important points of this project for the end user.

3.1. Results of the Initial User Questionnaire

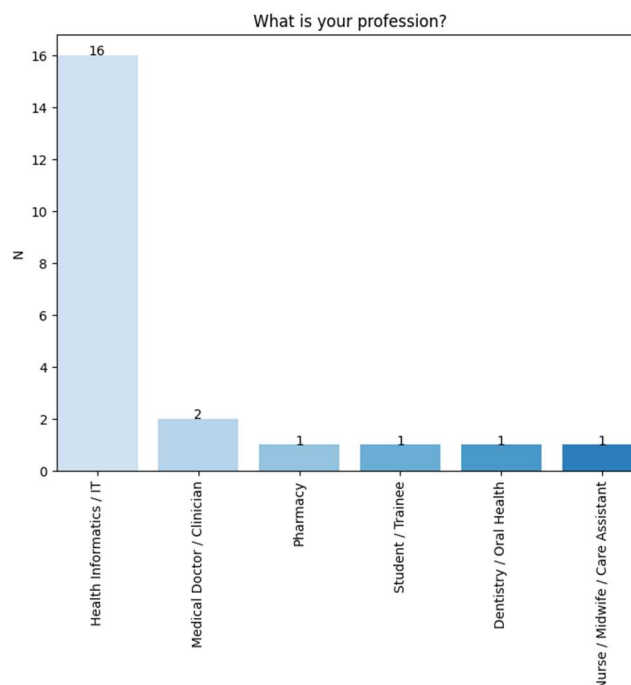
The initial questionnaire was distributed via LinkedIn, the openEHR newsletter, and the openEHR Discourse forum.

- **Number of responses:** 22
- **Response period:** 14/07/2025 - 28/08/2025

Section I: Getting to Know the Professional

Professional Background

Of the 22 participants, the majority (16 or 72.7%) work in Health Informatics or IT. The remaining respondents identified as clinicians, students or trainees, or worked in pharmacy or dentistry.

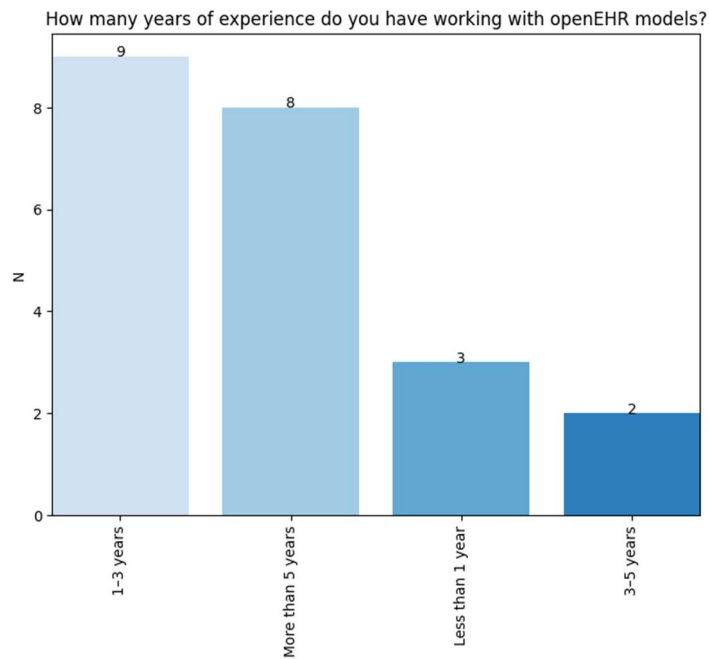


Distribution of professions among participants.

Conclusion: Although the sample size is small, the high proportion of participants from Health Informatics and IT suggests that primarily technical and informatics professionals currently drive the openEHR modelling community. However, the inclusion of clinicians and other healthcare professionals points to the interdisciplinary interest, which is essential for developing clinically relevant models.

Experience with openEHR Models

Most participants reported having 1-2 years of experience with openEHR models (9 respondents), closely followed by those with more than 5 years of experience (8 respondents). The rest had either less than 1 year or between 3 to 5 years of experience.

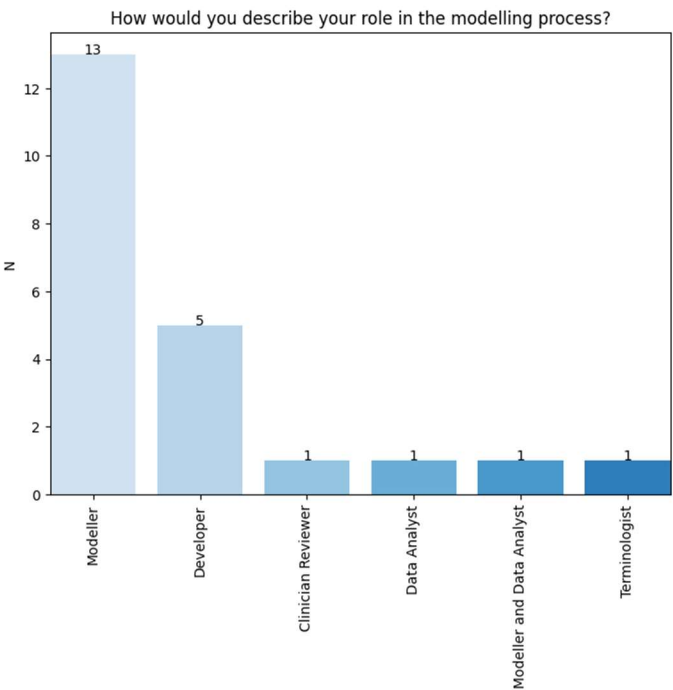


Years of experience with openEHR.

Conclusion: The distribution shows a mix of newcomers and highly experienced users, with fewer participants in the intermediate range. This may reflect a recent increase in interest and adoption, as well as a core group of long-term contributors. To address the experience gap and support community development, it would be beneficial to provide tailored resources for both predominant groups.

Roles in the openEHR Modelling Process

Most participants are involved as Modellers (13) or Developers (5) in the modelling process.

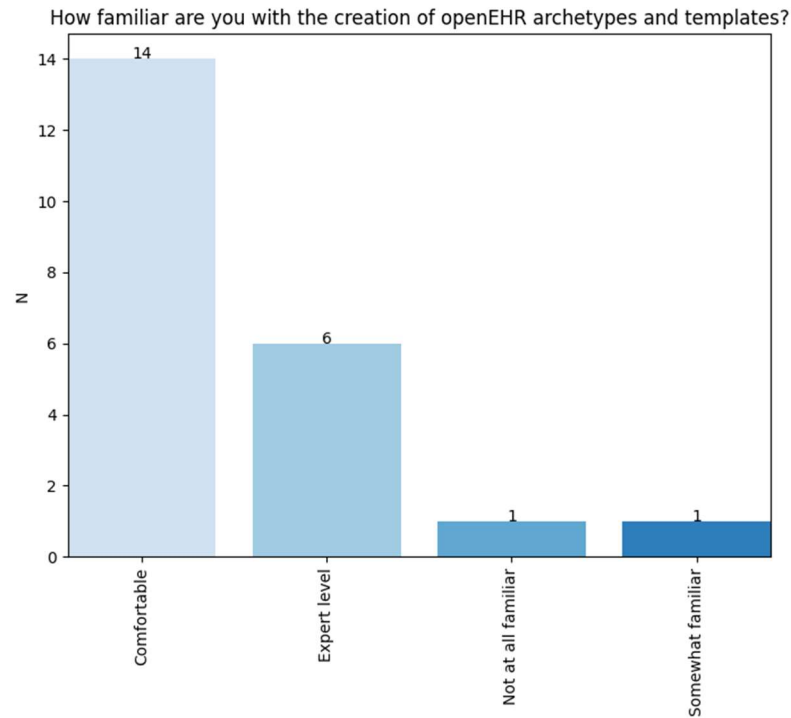


Roles in the openEHR modelling process.

Conclusion: The predominance of modellers and developers among participants aligns with previously reported professional backgrounds and underscores the technical orientation of current openEHR projects.

Familiarity with openEHR Artefact Creation

The majority of participants (14 out of 22) reported feeling very comfortable with the creation of openEHR artefacts, and 6 participants self-identified as experts in this area.



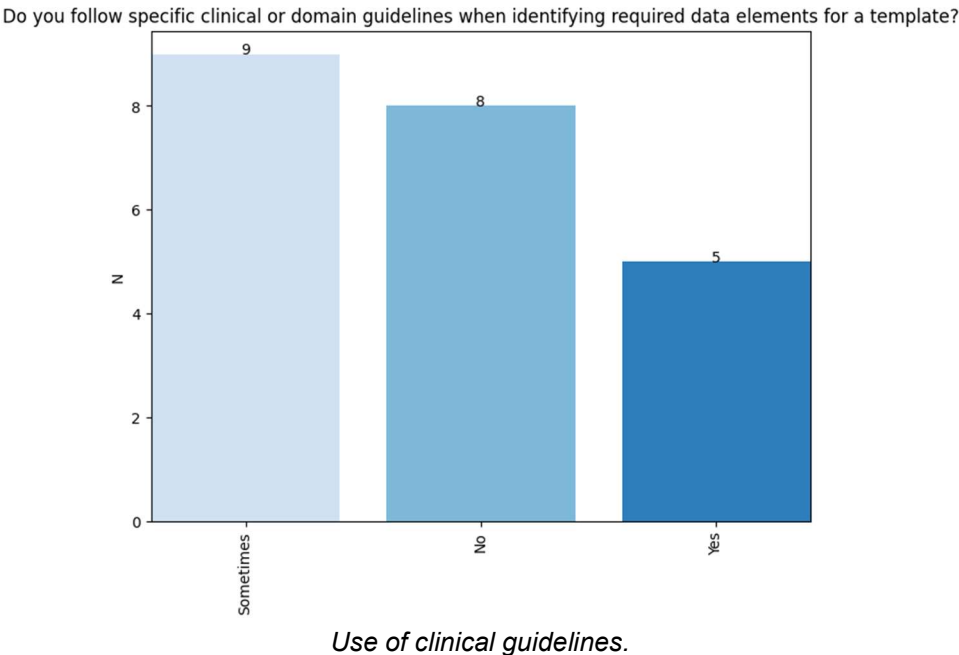
Familiarity with the openEHR modelling process.

Conclusion: The high level of comfort and expertise among participants corresponds with the previously reported experience levels (1–2 years or more than 5 years) of professionals, indicating a strong technical foundation within the community. To maintain inclusivity and encourage growth, it will be important to provide ongoing support and training opportunities for less experienced users.

Section II: Drafting and Compiling a List of Data Elements for an openEHR Template

Use of Clinical Guidelines in Template Creation

When asked about the use of clinical guidelines for template creation, 5 participants stated they definitely use such guidelines, while 9 reported using them only sometimes. Notably, 8 participants indicated that they do not use specific clinical or domain guidelines.



Conclusion: There is significant variability in the use of clinical guidelines during template creation. While some participants consistently apply guidelines, a substantial portion do not, which may affect the standardization and clinical relevance of developed templates. Promoting best practices and providing accessible guideline resources could help improve consistency and quality across the international community.

Guidelines and Resources Used for openEHR Template Creation

Of the 14 participants that reportedly use guidelines, 10 indicated the type of guidelines. The guidelines are grouped and listed here.

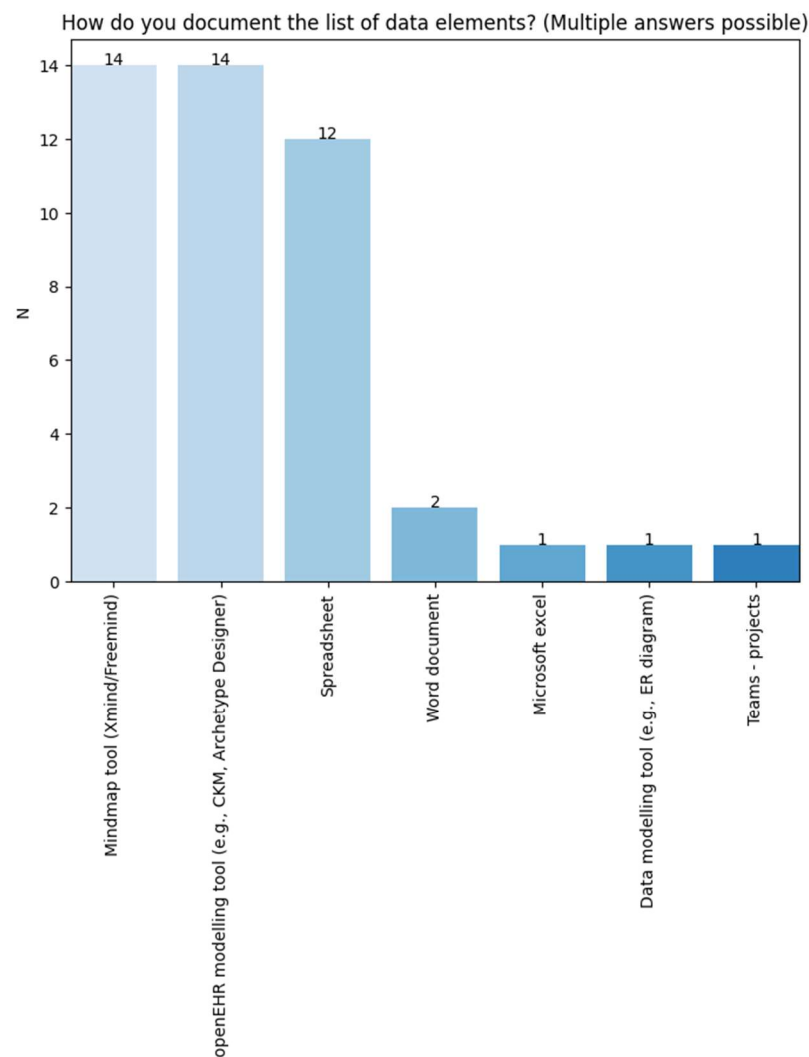
- Clinical and Domain-Specific Guidelines: Guidelines applicable to the specific domain or project at hand.
 - **Specific Clinical Guidelines:** Guidelines tailored to a concept or diagnosis, such as:
 - Patient-Reported Outcome Measures (PROMs) / PROM Styleguide: For the design and implementation of patient-reported outcome measures.
 - Assessment tools in psychiatry
 - Norwegian national guidelines for initial treatment of Non-muscle-invasive Bladder Cancer (NMIBC)
 - **Registry Specifications:** Requirements from clinical registries, often combined with clinician needs.
 - **Digitized Paper Forms:** Data elements are sometimes derived from existing paper forms, which may include coding systems like SNOMED-CT.
 - **Expert Domain Guidelines:** Guidance from domain experts, regional or national models.
 - **Customer-Provided Guidelines:** Templates based directly on whatever documentation or requirements the customer (e.g., hospital, laboratory) provides.
- International and National Standards
 - **FHIR Implementation Guides:** Used for interoperability and mapping between openEHR and FHIR resources.
 - **PRSB Guidance:** Professional Record Standards Body guidance for structuring clinical records.
 - **Clinical Guidelines (e.g., NICE):** Mentioned but considered rarely helpful by some.
 - **Standardized Specifications:** Such as ICHOM (International Consortium for Health Outcomes Measurement).
 - **SNOMED-CT and Other Coding Systems:** Used for coding data elements, sometimes requiring collaboration with clinical experts to identify appropriate codes.
- Technical specification
 - **UML Class Diagrams:** UML diagrams (used to visualize and structure data models and their relationships) of existing data models.

The creation of openEHR templates is highly dependent on a mix of clinical, technical, and project-specific guidelines. While technical and coding standards (like FHIR and SNOMED-CT) are widely used, domain-specific guidelines, registry requirements, and sometimes even digitized paper forms often shape the actual clinical content.

Platforms Used for Documenting Template Elements

When asked about the platforms used to document the list of elements for a template, 12 participants indicated they use two or more platforms. The three most commonly used platforms are:

- **Mind map tools** (14 participants)
- **openEHR-specific modelling tools** (such as CKM or Archetype Designer) (14 participants)
- **Spreadsheets (including Microsoft Excel)** (13 participants)



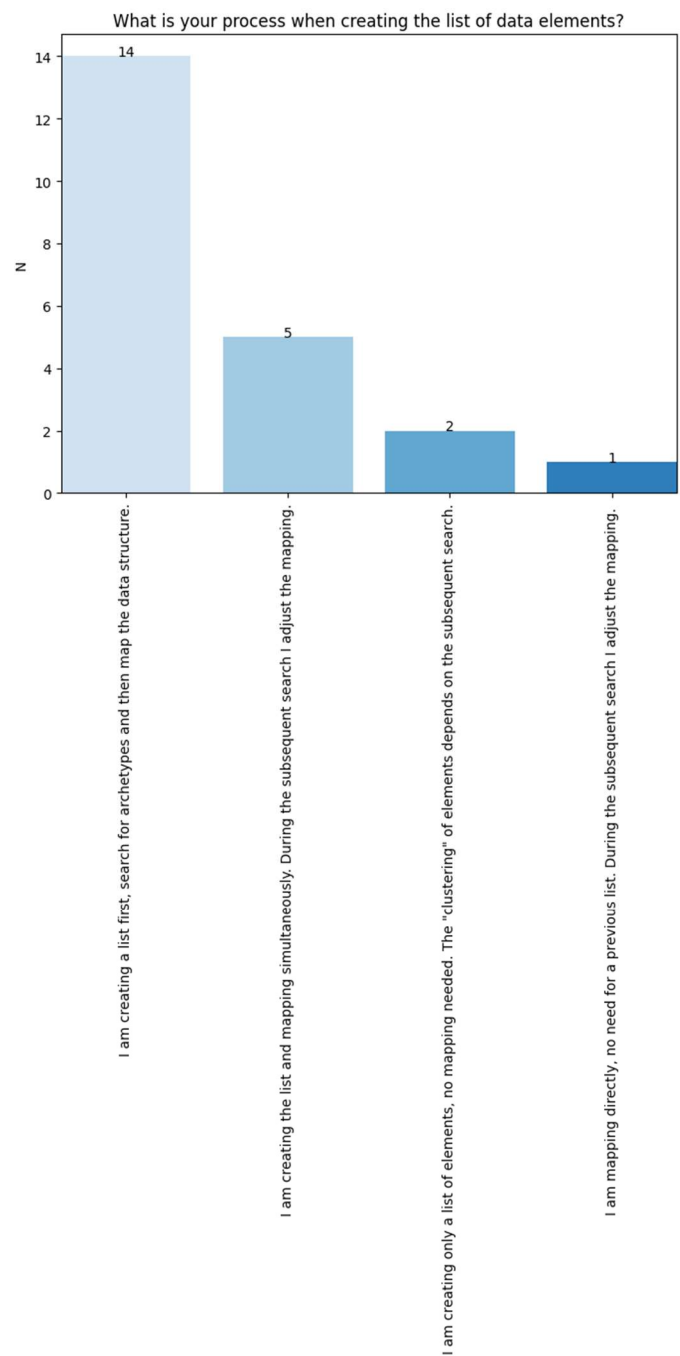
Platforms used to document the elements for the template.

Conclusion: The use of multiple platforms highlights a complex, but pragmatic approach to documentation. Mind maps and modelling tools are as popular as spreadsheets, suggesting that both visual and structured methods are valued for capturing and organizing template elements.

Process for Creating Lists of Data Elements

For the compilation of data elements necessary for the developed template, most participants (14) reported using a sequential process: first creating a list, then searching for archetypes,

and finally mapping the data structure. Five participants prefer a parallel approach, creating the list and mapping simultaneously.

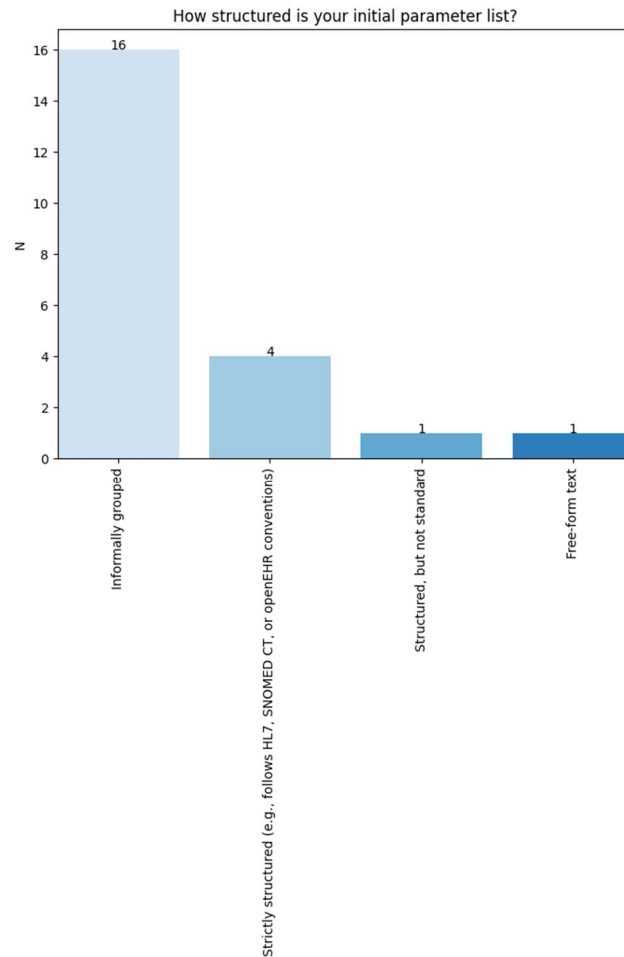


Process when creating list of data elements.

Conclusion: The sequential approach is dominant, likely because it provides a clear process. However, teams that value flexibility and iterative refinement during template development use parallel workflows.

Structure of Data Element Grouping

When documenting, most participants (16) group their data elements informally, while only 4 adhere to a strict structure.

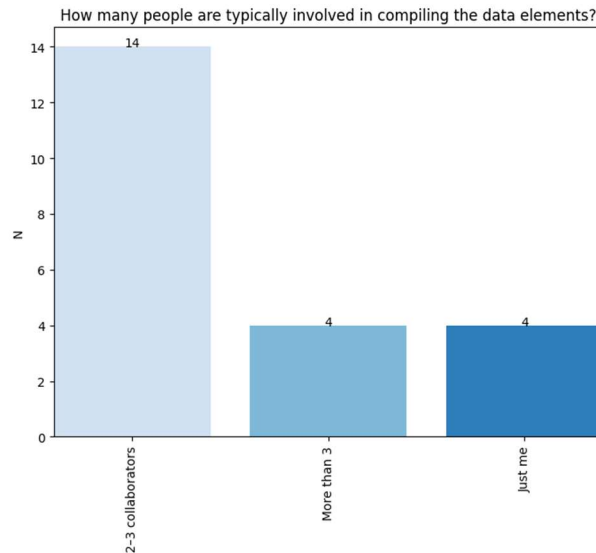


Level of data element structure.

Conclusion: Informal grouping is the norm, providing adaptability but also can lead to inconsistencies. Introducing structuring guidelines might help balance flexibility with standardization.

Team Involvement in Data Element Compilation

The group of people involved in collecting and structuring data elements is typically small, with 2-3 collaborators reported by 14 participants.



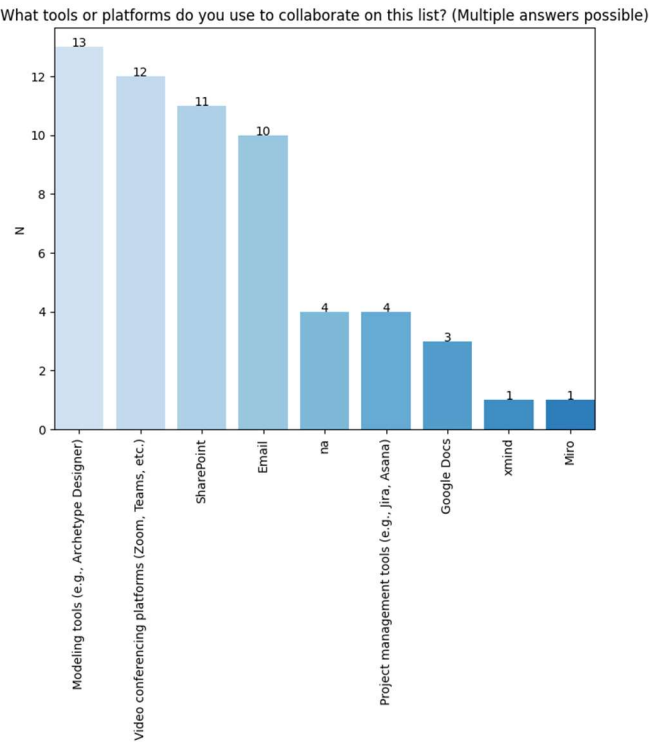
Involvement in the compilation of data elements.

Conclusion: Small teams are the standard, which streamlines communication and decision-making but may limit diversity of input. Encouraging broader collaboration could enhance the completeness of templates.

Tools Used for Collaboration

As reported by the participants, the main tools used for collaboration are:

- **Modelling tools** (e.g., Archetype Designer) (13)
- **Video conferencing** (12)
- **SharePoint** (11)
- **Email** (10)

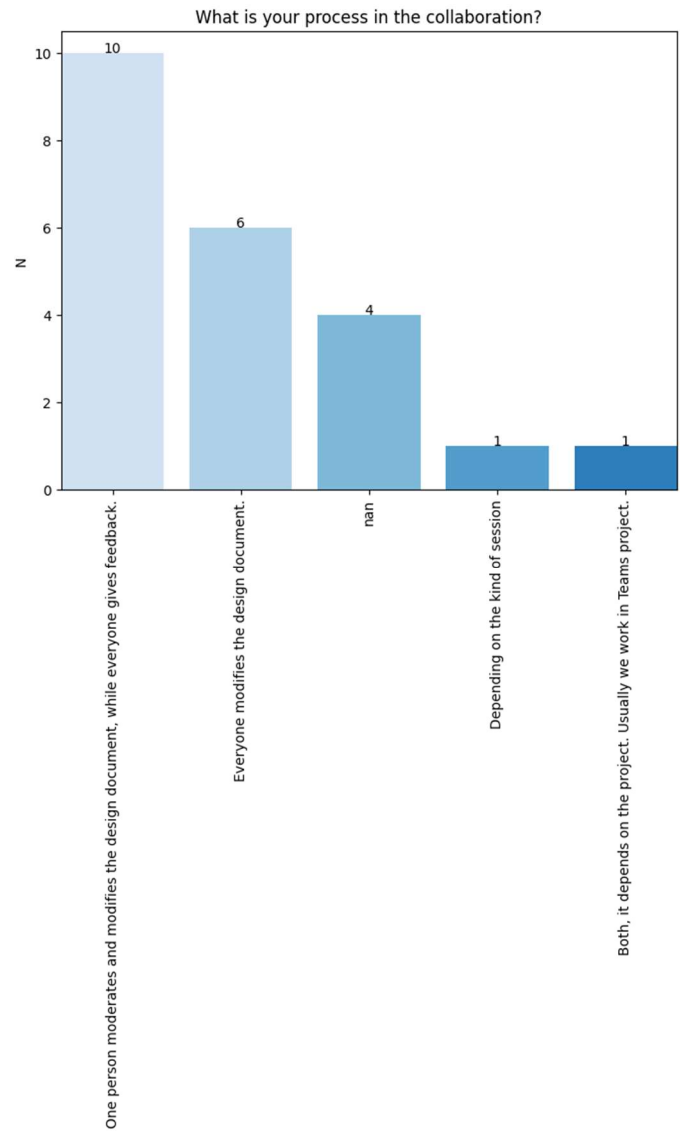


Tools used for collaboration in compilation of data elements.

Conclusion: Collaboration depends on a combination of specialized and general-purpose tools, reflecting the diverse range of platforms that data modellers need to be familiar with.

Moderation of Collaboration

One person who manages and modifies the design document, while all team members provide feedback typically moderates collaboration.



Moderation process in the compilation of data elements.

Conclusion: Centralized moderation helps maintain consistency and direction, but it is important to ensure that all voices are heard and that feedback is effectively integrated into the developed data model.

Expected Benefits of Collaboration in openEHR Template Modelling

14 of the 22 participants answered to the question about expected benefits from collaboration. The responses are grouped and summarized in the following list:

- **Diverse Perspectives and Knowledge Sharing**
 - Collaboration includes different viewpoints, increasing the depth of knowledge in both clinical and technical fields.
 - Involving clinicians and informaticians uncovers "silent knowledge" and ensures that information requirements not previously considered are identified early.
 - Participants learn about new tools and modelling approaches from each other.
- **Improved Data Quality and Error Reduction**
 - Working together helps catch mistakes early and enables comprehensive review, resulting in fewer errors.
 - Collaboration ensures higher data quality and improves the definition of information elements.
 - Multiple reviewers and perspectives help check for errors and optimize the final product.
- **Alignment with Real Clinical Needs**
 - Collaboration promotes shared understanding and aligns templates with actual clinical requirements.
 - Input from clinicians ensures that the logical data schema reflects real-world practice and needs.
 - Discussion of concepts, design, and coding systems with domain experts ensures correct use of codes and standards.
- **Increased Engagement and Investment**
 - Collaborative work makes participants feel more invested in the outcome and fosters a sense of ownership.
 - Streamlined modelling processes benefit both EHRs and case report forms, making the work more efficient and effective.
- **Process Optimization**
 - Collaboration helps streamline the modelling process, making it more efficient and effective.
 - Comprehensive input and review lead to the best possible modelling process and final product.

Conclusion: Participants expect collaborations to positively impact knowledge sharing, data quality, alignment with clinical needs, engagement, and process optimization.

Main Problems and Desired Changes in Compiling Data Element Lists

Of the 22 participants, 13 responded to the question: "What is the ONE most important problem that you have? What is the ONE change that you would like to see in the compilation of data element lists?" Their responses are grouped and summarized below.

- **Collaboration and Communication Challenges**
 - Difficulty staying on the same page, leading to messy or duplicated work.
 - Lack of a clear, shared system or tool for tracking contributions and changes.
 - Limited collaboration: clinical teams may see template design as too technical and distant, resulting in insufficient involvement.
 - Consensus is hard to achieve, especially when stakeholders have different backgrounds or priorities.
 - Desire for better collaboration tools and easier sharing of template views for client feedback.
 - Technical limitations, such as SharePoint not supporting Xmind documents and restricted access/versioning in Archetype Designer.
- **Knowledge and Understanding Gaps**
 - Lack of understanding of the meaning and structuring of information elements among collaborators, especially clinicians.
 - Clinicians often do not know how to structure or cluster data, making it difficult to create robust templates.
 - Need for greater awareness that templates should be consensus-driven, not tailored to individual needs only.
 - Desire for more education on how openEHR works to increase meaningful involvement.
- **Process and Content Issues**
 - RM (Reference Model) fields are often missed during element compilation.
 - Many required data elements do not have a clear mapping to existing archetype elements, especially when they infer knowledge (e.g., "Oncological history: yes/no").
 - Lack of guidelines, implementation guides, or best practices from openEHR.
 - Desire for an AI-generated draft to suggest fitting archetypes and elements at the start of the process.
- **Tooling and Technical Limitations**
 - Difficulty with versioning and access in modelling tools (e.g., Archetype Designer requires sharing usernames/passwords, lacks easy multi-user access and versioning).
 - Desire for improved collaboration tools and easier ways to share and comment on templates.

Conclusion: Participants reported that effective collaboration in compiling data element lists is hindered by both social and technical barriers. Key problems include a lack of user-friendly, centralized platforms for collaboration, insufficient interdisciplinary understanding, and a significant knowledge gap, especially for clinicians, regarding template creation. Additionally, participants noted inadequate guidance, limited automated support for mapping and element selection, and insufficient documentation of best practices. Outdated or inadequate technical infrastructure and tools were also identified as obstacles to seamless and secure collaborative workflows.

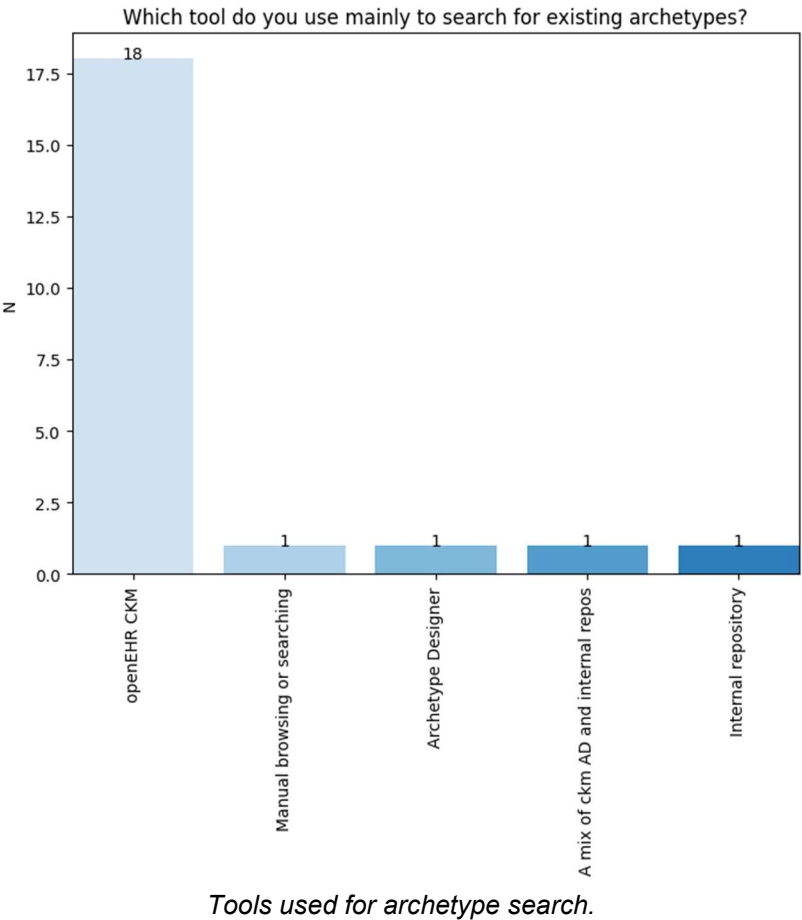
To address these issues, it is suggested introducing user-friendly, centralized collaboration platforms, fostering interdisciplinary understanding, providing targeted training and clear

guidance for clinicians, improving guidance and documentation, automating mapping and element selection, and upgrading technical infrastructure and tools.

Section III: Searching Archetypes

Archetype search tools

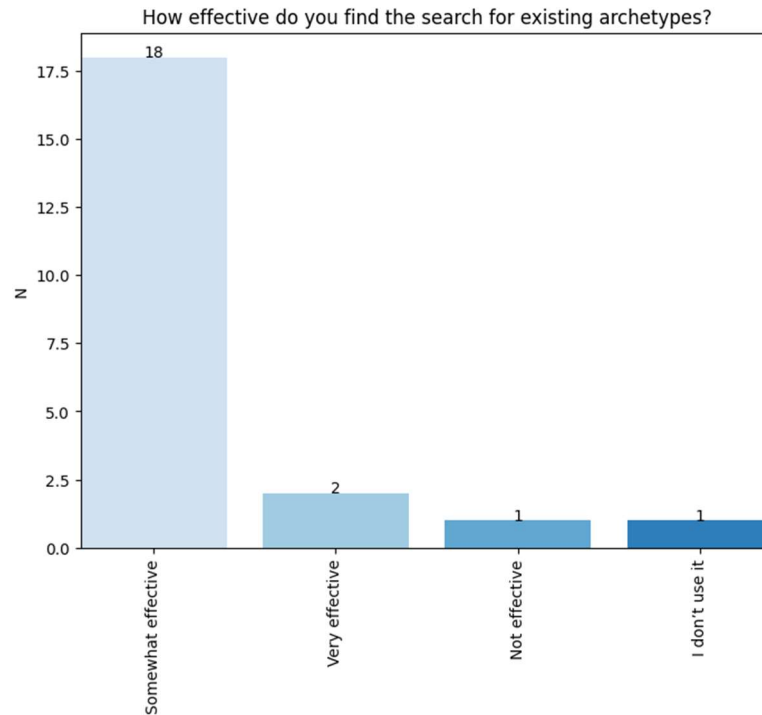
The vast majority of participants (81.8%) rely on the openEHR Clinical Knowledge Manager (CKM) as their primary tool for searching appropriate archetypes for templates.



Conclusion: Most users depend on CKM as their main archetype search tool.

Effectiveness of search tools

Most participants (81.8%) rate the search process as only somewhat effective, indicating room for improvement in the current tools and methods.



Effectiveness of the search tools.

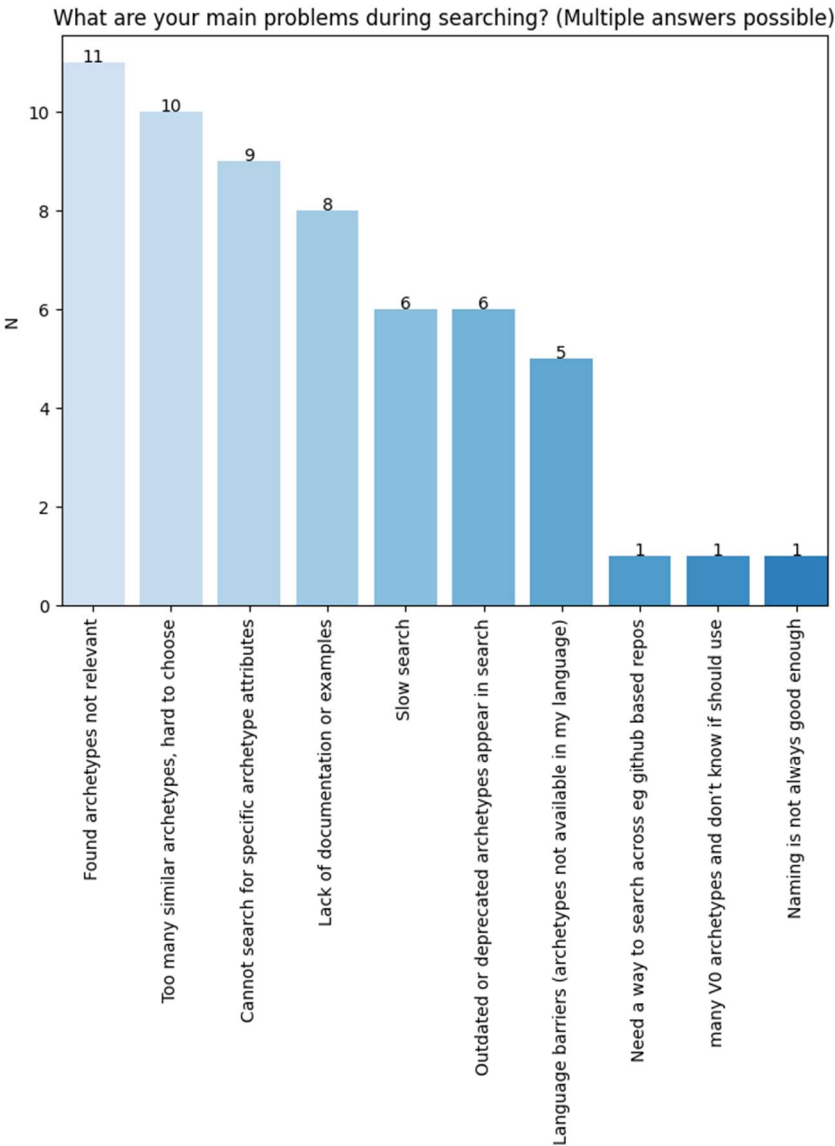
Conclusion: The majority find current search tools only somewhat effective, which indicates a need for better tools.

Main Challenges in Archetype Search

The main challenges reported by participants are summarized below:

- Inclusion of irrelevant archetypes in search results
- Too many similar archetypes, making selection difficult
- Lack of search functionality for specific attributes
- Insufficient documentation or practical examples

Secondary issues include slow search performance, outdated archetypes appearing in results, and language barriers.



Main problems encountered during archetype search.

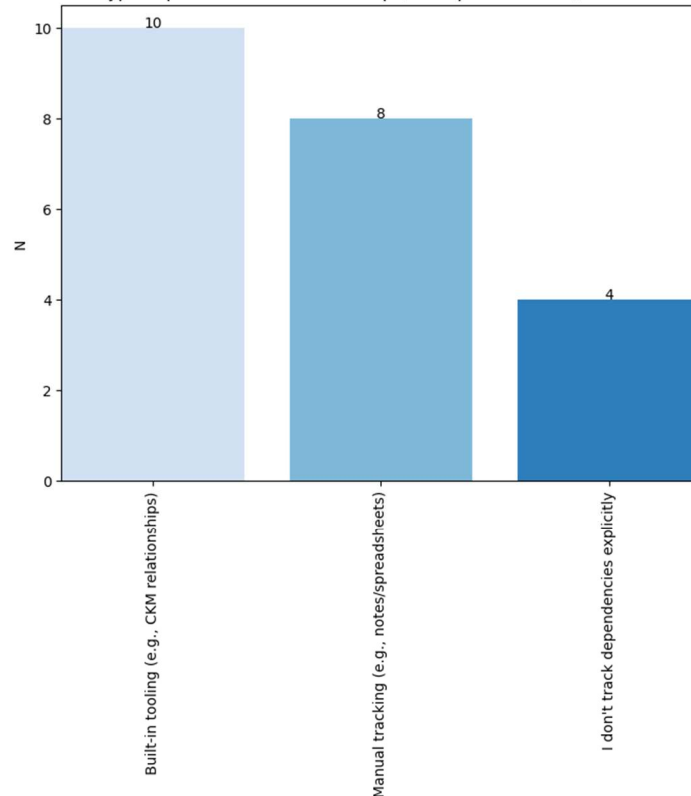
Conclusion: Users primarily face challenges such as irrelevant results, similar archetypes, and lack of attribute-based search. This also indicates a need for better search tools with better adjustment to user needs.

Tracking Archetype Relationships

Participants use a variety of methods to track archetype relationships:

- 4 participants do not track relationships at all
- 10 use built-in tools (e.g., CKM features)
- 8 track relationships manually

How do you keep track of archetype dependencies or relationships, like specializations, inclusions or exclusions, during planning?

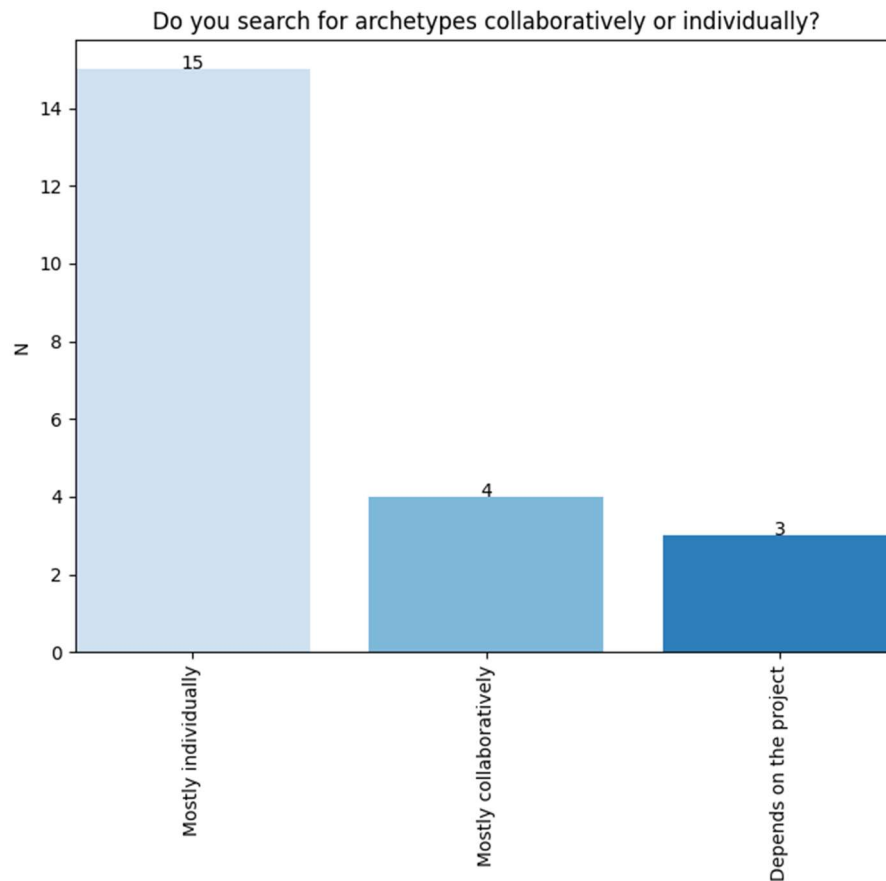


Methods for keeping track of archetype relationships.

Conclusion: Most participants use built-in tools or manual methods to track archetype relationships, while some do not track them at all. The interdependency of archetypes seems to be sufficiently documented by the existing tools (CKM).

Collaboration in Archetype Search

The search process is predominantly individual, with 15 out of 22 participants searching alone. Only 4 participants reported mostly collaborative searches.

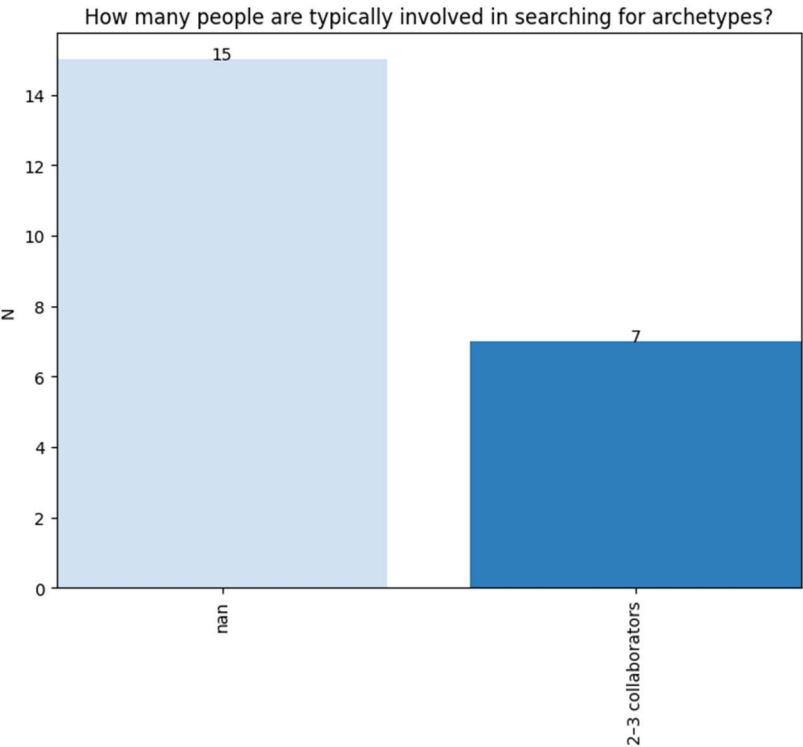


Collaborative versus individual archetype search.

Conclusion: Archetype search is usually performed individually rather than collaboratively. There does not seem a need for a collaborative tool.

Team size in search collaboration

When collaboration occurs, it typically involves small teams of 2-3 people, as reported by 7 participants.



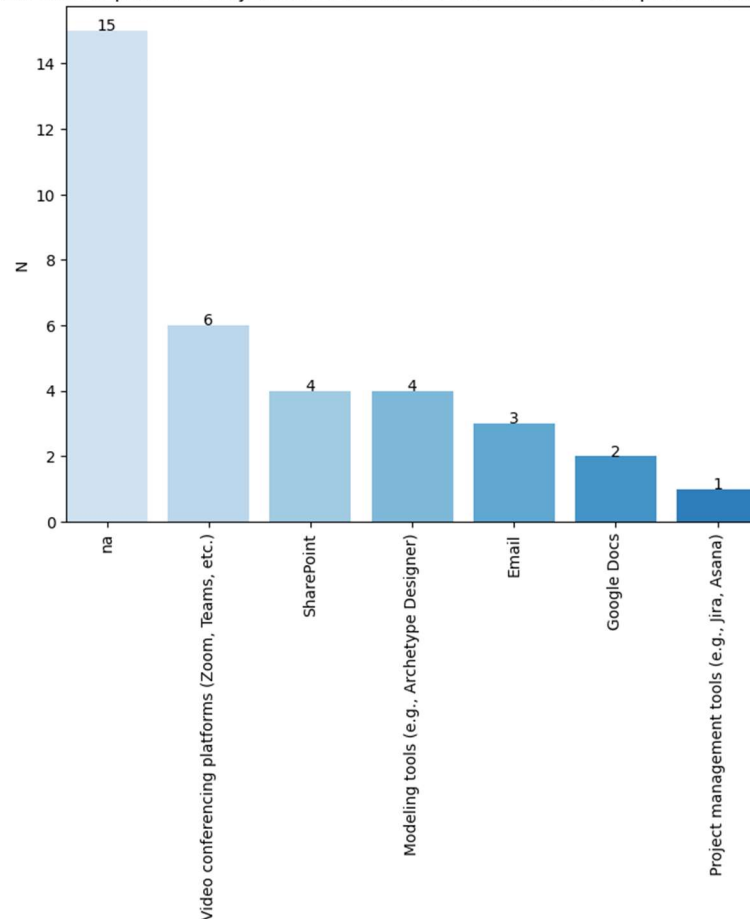
Typical team size for collaborative archetype search.

Conclusion: When collaborative archetype search takes place, it typically involves small teams of 2-3 people. Therefore, any collaborative tool designed for this purpose can remain lightweight and does not need to support large-scale collaboration.

Platforms used for collaborative search

Although only 7 participants provided feedback on the platforms used for collaborative search, the most common tools mirror those used for compiling data elements: modelling tools (e.g., Archetype Designer), video conferencing, SharePoint, and email.

What tools or platforms do you use to collaborate in the search? (Multiple answers possible)



Platforms used for collaborative archetype search.

Conclusion: The main platforms for collaborative search are modelling tools, video conferencing, SharePoint, and email.

Main Expected Benefits of Collaborative Archetype Search

Six participants provided responses regarding the main benefits they expect from collaboration during archetype search. Their answers can be grouped as follows:

- **Improved Quality and Suitability:**
 - Finding better-quality archetypes through shared knowledge
 - Selecting archetypes that best fit the information elements and specific project needs
 - Ensuring all necessary elements are included and nothing important is missed
 - Identifying the most optimal archetype for a given use case or topic
- **Efficiency and Speed:**
 - Saving time by leveraging the collective expertise of the group
 - Achieving greater speed and accuracy in archetype selection

Summary: Collaboration in archetype search is valued by participants for enhancing the quality and relevance of selected archetypes, ensuring completeness, and increasing the efficiency of the process. By working together, participants can pool their knowledge, avoid overlooking important elements, and make faster, more accurate decisions.

Main Problems and Desired Changes in Archetype Search

Twelve participants provided responses to the question about the most important problem they face and the one change they would like to see in searching for archetypes. Their answers are grouped as follows:

- **Search Precision and Filtering:**
 - Difficulties in quickly finding the right archetype due to too many similar or irrelevant results
 - Desire for advanced filters, tagging, or structured search options (e.g., by document type, compilation type, or element level)
 - Need for more synonyms and better search on element level
 - Request for a better general index or overview of available archetypes, not just textual search
 - Desire for sorting options (e.g., by date in project/incubator CKM)
- **Clarity and Semantic Consistency:**
 - Confusion due to redundancy between main archetypes and included archetypes, leading to inconsistent modelling across organizations
 - Need for more semantic links between data elements (not just coded values)
 - Desire for mapping information elements to archetype elements by label, not just meaning
- **Tooling and Process Improvements:**
 - Need for tooling-neutral searching
 - Desire for a more effective process for submitting and handling change requests

Summary: As indicated by the participants, the most pressing issues in archetype search are the lack of precise filtering and advanced search options, leading to difficulty in finding the right archetype efficiently. Participants also highlighted the need for improved clarity, semantic consistency, and better tooling support. Addressing these challenges especially by enhancing search precision, providing better indexing and filtering, and improving semantic clarity would significantly improve the archetype search experience for users.

Optional Final Section: Suggestions and Feedback

Key Suggestions for Improving the modelling and Planning Process

Eight participants responded to the question: "If you could change one thing about the current modelling and planning process, what would it be?". Their answers are grouped as follows:

- **Flexibility and Adaptability:**
 - Make archetypes more flexible to accommodate changes without technical issues.
 - Integrate AI-based tools to assist in selecting the right archetypes.
- **Collaboration and Stakeholder Involvement:**
 - Involve clinicians early and often to ensure models reflect real-world needs.
 - Promote more structured and well-organized international collaboration, despite differing priorities and projects.
 - Better inform and empower users to contribute to improving information models, not just use them passively.
- **Process and Methodology Enhancements:**
 - Provide more structured reasoning and documentation for modelling decisions.
 - Integrate openEHR modelling more closely with FHIR implementation guides.
- **Terminology and Semantics:**
 - Give greater weight to terminology (e.g., SNOMED CT) so that semantics drive modelling, regardless of where a data element appears.

Conclusion: Participants emphasized the need for greater flexibility, better stakeholder engagement (especially clinicians), improved international collaboration, and stronger integration of terminology and standards. Addressing these areas could make the modelling and planning process more effective, inclusive, and aligned with real-world clinical needs.

Suggestions for Tools or Features to Improve Workflow with openEHR Models

Ten participants provided suggestions for tools or features that would enhance their workflow with openEHR models. Their responses are grouped as follows:

- **Search, Impact Analysis, and Change Management:**
 - Support for AQL (Archetype Query Language) impact analysis and editing when archetypes or templates change.
 - More effective and timely change request processes, especially for international collaboration.
 - Ability to sort by date in projects and incubators; fix bugs in CKM to Archetype Designer downloads/uploads.
- **Tool Integration and Usability:**
 - Closer integration between Archetype Designer and CKM.
 - Expose and leverage the full capabilities of the Archetype Designer API.
 - Tooling improvements to make RM (Reference Model) fields more visible, especially for newcomers.
 - Convert templates to ASCII documents in Archetype Designer with all elements included.
- **Terminology and Semantic Support:**
 - Better correspondence and integration between openEHR models and terminologies such as SNOMED CT for more semantic searches.
 - Each data element should have a semantic link to terminology, avoiding reliance on internal codes.
 - More reference templates to guide modelling.

Conclusion: The most desired improvements focus on better change management, enhanced tool integration, and stronger semantic support. Streamlining workflows, making tools more user-friendly, and ensuring robust links to clinical terminologies would significantly improve the efficiency and quality of working with openEHR models.

3.2. Key Findings of the Questionnaire

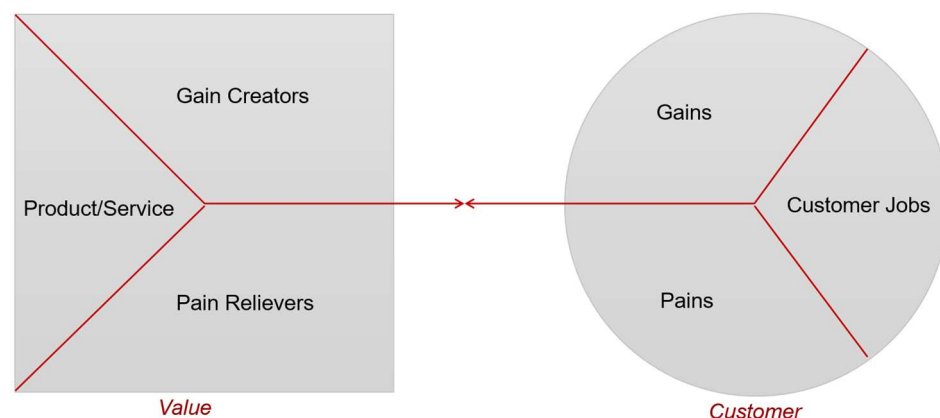
- **Guidelines and Tools:** Participants use a wide range of guidelines for template creation, including clinical, domain-specific, and project-specific sources. The most common tools for documenting elements are mind maps, spreadsheets, and openEHR-specific modelling tools (CKM, Archetype Designer).
- **Archetype Search Practices:** The openEHR CKM is the primary tool for searching archetypes (used by 81.8%). Most users find the search only somewhat effective, citing issues like irrelevant or redundant results, lack of advanced filters, and insufficient documentation.
- **Main Search Challenges:** Key problems include difficulty filtering/searching by specific attributes, too many similar archetypes, outdated results, and language barriers. Users want better indexing, more synonyms, and element-level search.
- **Tracking Relationships:** Approaches vary: some use built-in tools, others track manually, and a few do not track relationships at all.
- **Collaboration:** Archetype search is mostly done individually, but when collaboration occurs (typically 2-3 people), it is valued for improving quality, completeness, and speed of selection. Collaboration platforms include modelling tools, video conferencing, SharePoint, and email.
- **Desired Improvements:** Users want more flexible archetypes, better integration with standards (e.g., FHIR, SNOMED CT), improved change request processes, and AI-based tools for archetype selection. There is a strong call for more structured international collaboration and greater clinician involvement.
- **Tooling and Workflow Suggestions:** Participants request better impact analysis for AQL, improved tool integration (especially between CKM and Archetype Designer), more visible RM fields, bug fixes, and enhanced semantic support (e.g., semantic links to terminology).

4. Value Proposition

With the context of the proposed project and the feedback from the questionnaire in mind, a value proposition was created before going on with the project.

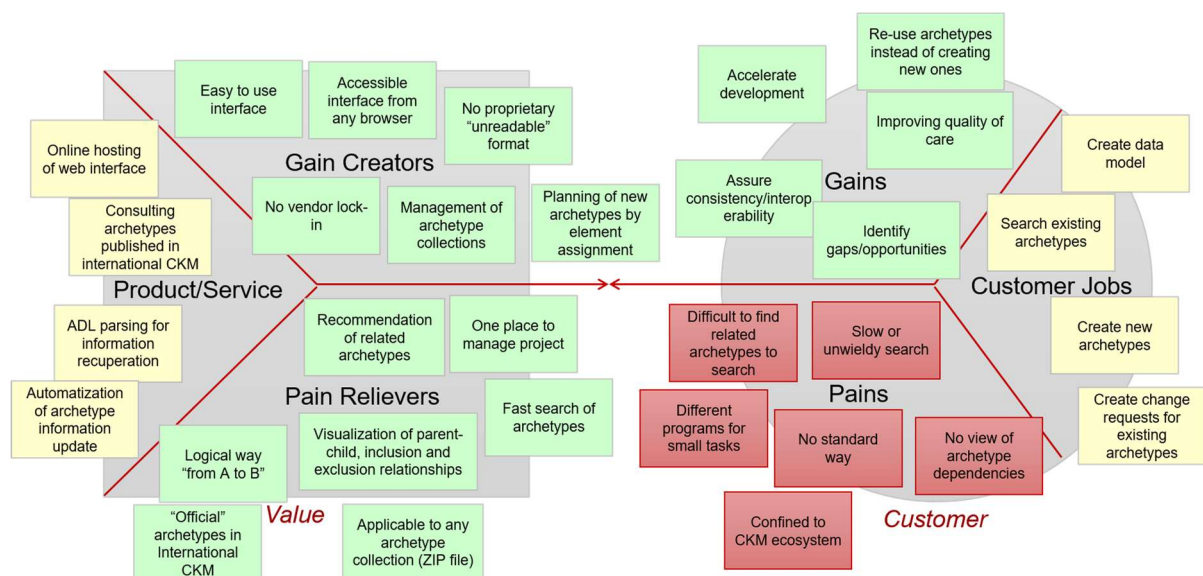
This proposition was developed by using the **Value Proposition Canvas** from Strategyzer (<https://www.strategyzer.com/library/the-value-proposition-canvas>).

The Value Proposition Canvas helps systematically understand and address the needs of stakeholders. It works by mapping out two key areas: the "Customer Profile" (which identifies stakeholder jobs, pains, and gains) and the "Value Map" (which outlines how a solution's products and services relieve pains and create gains). By identifying stakeholders' pain points and desired gains, the canvas enables to target proposed solutions at the clinical knowledge modellers' needs.



Value Proposition Canvas

After completing the canvas, the following points can be concluded for the "customer" side (the clinical knowledge modeller) and the "value" side (the application to be developed).



The completed Value Proposition Canvas

4.1. The "Customer" side

The clinical knowledge modeller wants to create a data model structure, based on a list of needed data elements. For this, the modeller has to:

- Search and identify "official" existing archetypes that include fitting data elements
- Identify possible necessary changes in existing archetypes to accomodate needed data elements
- Identify the need for new archetypes that include data elements that cannot be found or included in existing archetypes

The most important gains for the modeller in determination of the data model structure are:

- **Re-usability:** By re-using existing archetypes, the This leads also to a faster development.
- **Interoperability:** The use of existing archetypes result in a higher interoperability between different systems.
- **Identify gaps:** The model structure can reveal archetypes that have not been developed by now in the openEHR community.
- **Quality:** Ultimately this approach results in a better quality in healthcare, by using interenationally developed models.

The most hindering factors for the modeller in this undertaking are:

- **No alternative search engines:** The search for archetypes is contained to the CKM by now. The modeller is limited to few or no alternative search engines.
- **Search for related archetypes:** Starting from an archetype, it is difficult to find related archetypes or to visualize the relation to other archetypes.
- **Fragmented workflow:** The workflow of creating a data structure contains various programs (Excel, XMind, etc.) which are not specific for openEHR and information has to be treated manually.

4.2. The "Value" side

To add to the modellers' gains and to remedy and overcome pain points, special attention will given to:

- Provide an **easy to use interface** for the webpage (search, planning and visualization tool).
- **No vendor lock-in for the input.** The data analysed can come from any ZIP file source with ADL1.4 archetype files.
- One project oriented software that can **manage a custom collection** of archtypes, including the assignment of elements to archetypes in the collection.
- **Visualization** of the archetype dependencies (parent-child, inclusion, exclusion, etc.).
- **Fast search** experience thanks to the lightweight archetype database.
- **Recommendation** of related or similar archetypes, based on a semantic recommendation algorithm.
- **No vendor lock-in for the output.** All output files (project save files, archetype data file, etc.) are going to be created in a readable format (such as JSON).

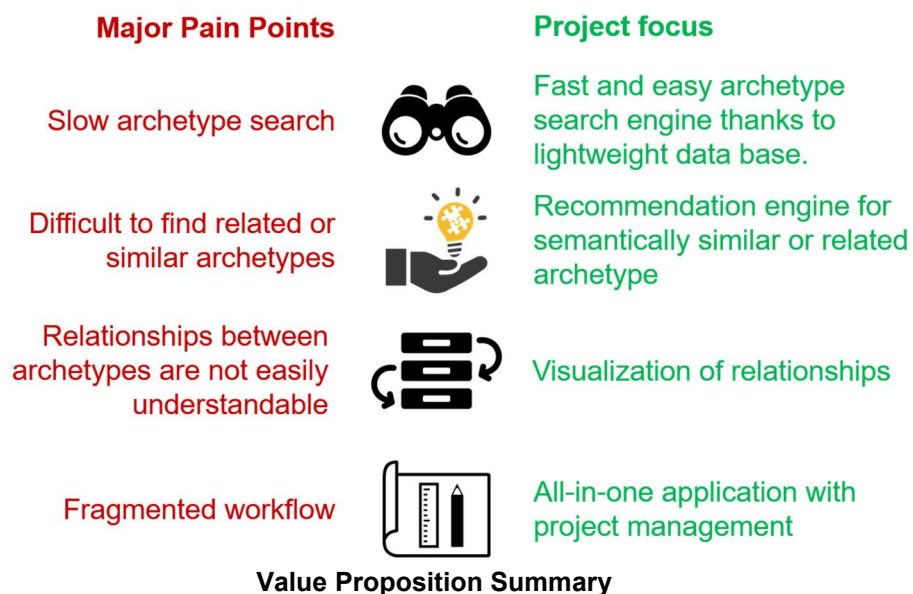
The services that will be needed to respond to the modellers needs are the following:

- **Consultation of archetypes** in the international CKM or any ZIP file source.

- **Parsing of ADL1.4 archetype** files for the needed information.
- Automatization of parsing for **periodic update** of archetype information.
- **Online hosting** of the webpage for searching, planning and visualization of archetypes.

4.3. Summary and formulation of the value proposal

Taking into account all the pains and gains, we can summarize the main pain points and the resulting focus of this project:



To put it into a short value proposition:

The aim of this project is to empower clinical knowledge modellers to efficiently search, visualize, and plan with openEHR archetypes through an intuitive, vendor-neutral tool. By streamlining the workflow and enhancing discoverability, it accelerates model development, improves interoperability, and supports higher quality healthcare data models.

5. Project planning

5.1. Research, Development, and Implementation Strategies

The primary objective of this project is to create a stable working prototype that effectively demonstrates value to the end user. To achieve this, the following methodology for research, development, and implementation is going to be used.

Research

The research phase involves conducting a questionnaire aimed at assessing the need for the proposed project and identifying users' pain points. This step is crucial as it helps to understand the specific requirements and challenges faced by potential users, ensuring that the final tool is tailored to meet their needs.

Development

The development process is divided into two main parts:

- **Part 1:** This part focuses on creating a tool designed to acquire archetype ADL files from either a local or online repository. The tool will extract relevant data and store it in a data file, facilitating easy access and management of the information.
- **Part 2:** The second part consists in developing an online search, visualization and planning tool. This tool will enable users to explore the previously generated data file, providing a user-friendly interface for data interaction and analysis.

Implementation

The implementation phase includes several key activities:

1. **Extraction of Archetypes:** Archetypes can be extracted from online resources and local zip files. The online repositories include CKM or GitHub repositories. This flexibility ensures that the tool can be applied to a wide range of archetype sources.
2. **Development of the Extraction Tool:** The extraction tool will be developed in Python and compiled into an executable (.exe) file with a graphical user interface. Additionally, a "headless" mode is going to be implemented for an automatic update of the resulting data file. Both the source code and the executable will be published in a GitHub repository under the Apache 2.0 open-source license, making them freely available to the community.
3. **Search, Visualization, and Planning Tool:** This tool will be accessible to any user with an internet connection and a modern browser. HTML and JavaScript are going to be used to develop a user-friendly interface. Data visualization is going to be implemented with d3.js, a widely-used library with comprehensive documentation available at <https://d3js.org/>. The tool will also be published in a GitHub repository under the Apache 2.0 open-source license.
4. **User Data and User Management:** All data that is used by the user is going to be saved on their local hard drive. No user data is going to be collected or sent to third parties. There are no user accounts or user management planned.
5. **User-Friendly Data Management:** A special effort is going to be made to ensure that data input and output are as user-friendly as possible. Users will be able to upload data in easily manageable formats, such as .xlsx, simplifying the process of data interaction.

6. **Maintenance and Transition:** Martin Koch from the Catalan Healthcare Service will maintain both tools in a public GitHub repository during the duration of the project. Upon completion of the project, there is potential for the content to be transitioned to an official openEHR domain. The future development and maintenance has to be discussed once the value of the project has been assessed.

Engagement with openEHR Specifications

This project aligns with the openEHR specifications by using and parsing archetype files that comply with the ADL standards. By adhering to these specifications, we ensure that our tools are compatible with the existing standard and can be effectively utilized within the broader openEHR community. Furthermore, the tools will utilize REST API calls to various CKM sources for retrieving archetypes.

Specific openEHR Tools, Platforms, or Clinical Modelling Approaches

The primary source of the archetypes integrated into this project will be the international CKM. The tools to be developed are going to integrate into the common modelling approach, where a list of data elements is given in the beginning, followed by a search for archetypes and planning a layout and finally creating a template. The main focus here lies on the searching and planning.

Quality Assurance

To assure a good quality of this project, the robustness of the data extraction tool will be rigorously tested against all available archetypes in the CKMs, including those from international, Norway, Catalan, and German sources. This comprehensive testing approach will help identify and rectify any issues, ensuring that the tool performs reliably across various scenarios.

Additionally, the quality of the visualization tool will be ensured through user feedback in real use cases. This feedback will be invaluable in refining the tool's functionality and user experience.

5.2. Timing by quarters

Quarter 1 (Months 1-3)

Activities

- Design and distribute initial questionnaire.
- Conduct stakeholder interviews via questionnaire (developers, clinicians, etc.).
- Analyse interview responses to extract needs and pain points.
- Define features for both tools (must-have, nice-to-have).
- Research existing tools/libraries for parsing ADL files.
- Research existing archetype browser or searching tools
- Develop initial version of the ADL extraction tool.
- Define data model and schema for archetype representation.

Deliverables

- Full feature list (categorized: essential / optional).
- Stakeholder interview analysis report.
- Technical specification document for the extraction tool.
- Architecture overview of the extraction tool (inputs/outputs/data flow).

Milestones

- Stakeholder needs assessment complete.
- First working prototype of ADL extractor (alpha).

Quarter 2 (Months 4-6)

Activities

- Finalize and publish the ADL extraction tool.
- Design and begin implementation of the web-based visualization/search/planning tool.
- Define UX/UI for visualization/planning tool.
- Set up service for updating parsed archetypes data file automatically.

Deliverables

- Fully functioning archetype extractor.
- Documentation for the ADL extractor.
- GitHub repository with source code.
- Wireframes/mockups for visualization tool.
- Basic front-end prototype for searching and visualizing archetypes.

Milestones

- Extraction tool publicly released.
- Search tool can manage archetype and project information.
- Technical specification document for the search and visualization tool.

Quarter 3 (Months 7-9)

Activities

- Finalize visualization/planning tool.
- Launch test version of the visualization/planning tool.
- Recruit pilot user group (developers, clinicians) for user feedback.
- Collect use cases for archetype creation and search.
- Design and distribute usability questionnaire.

Deliverables

- Test version of the visualization and planning tool.
- User guide and instructions.
- Use case compilation document.
- User feedback report from pilot test.

Milestones

- Searching/visualization tool released for testing.
- Real-world use cases documented.
- User experience feedback collected and analyzed.

Quarter 4 (Months 10-12)

Activities

- Implement final improvements based on user feedback.
- Finalize full documentation.
- Announce final public release.
- Conduct final presentation/demo for stakeholders.

Deliverables

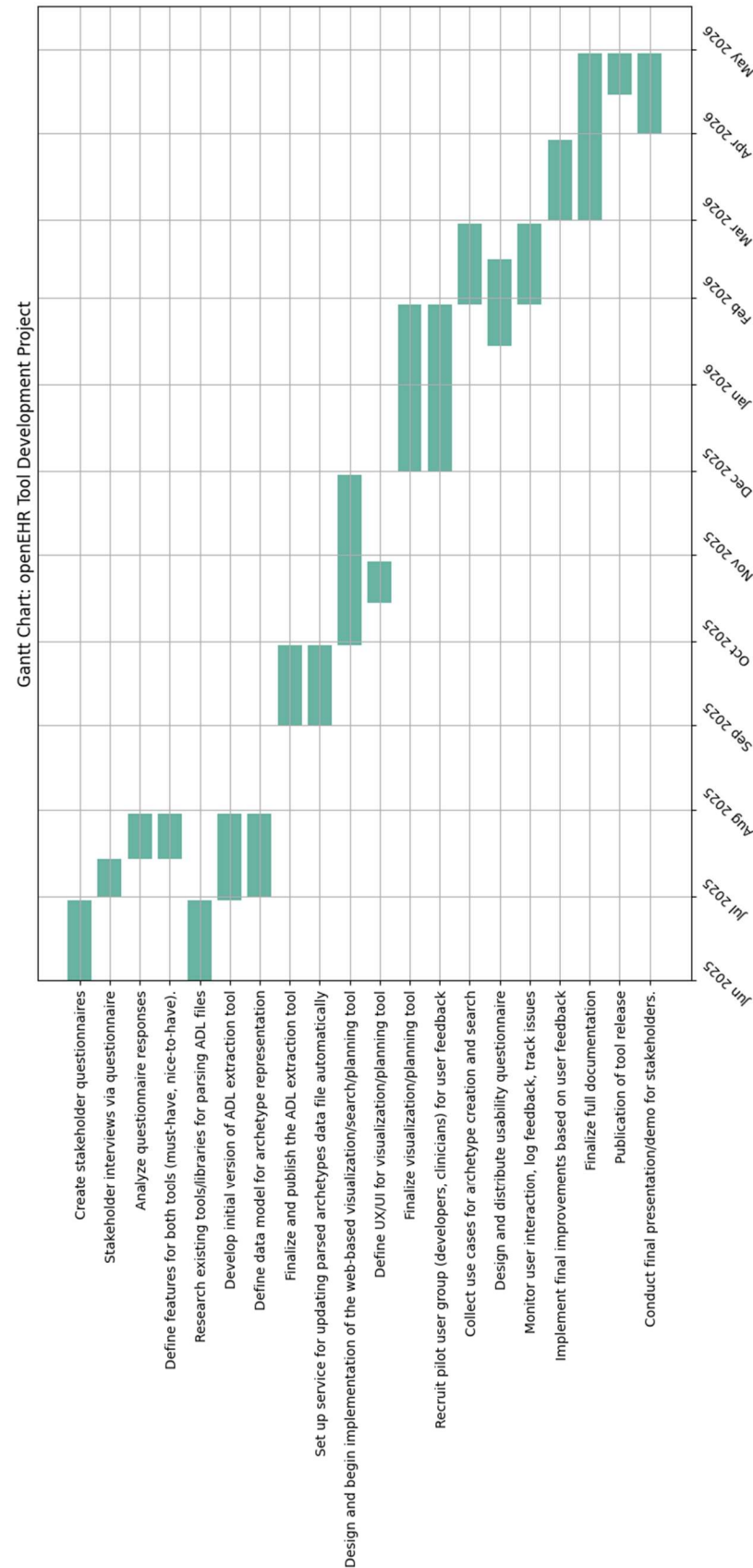
- Finalized visualization and planning tool.
- Complete documentation (installation, user manual, technical guidevisualization).
- Summary report of project progress, outcomes, and lessons learned.
- Final code repositories.

Milestones

- All user feedback processed.
- Tools packaged and documented.
- Project completed and reviewed with stakeholders.

5.3. Detailed timing

The timing by activities is laid out in the following Gantt chart.



6. Scope

6.1. In Scope

- Development of a data extraction tool for a collection of archetypes in ADL 1.4 format.
- Creation of an online search, visualization, and planning tool.
- Both tools are going to be stable and usable prototypes.
- Online hosting of the project throughout the duration of the fellowship.
- Archetype data is retrieved and generated periodically (e.g., once a day or once a week).
- Only applicable to ADL 1.4.
- All user data is saved locally.
- Data extraction in English or the original language if English is not available.

6.2. Out of Scope

- This project will not result in a finished product.
- There will be no creation of fully “fleshed-out” new openEHR archetypes.
- No development of openEHR templates.
- The project will not provide one comprehensive tool that combines extraction with search, visualization, and planning.
- Hosting and maintenance of the tools is not guaranteed beyond the conclusion of the fellowship.
- No real-time connection to a CKM or other archetype repository.
- No user account management.
- No ADL 2.0 compatibility.
- Does not support multiple languages.

7. Ethical Considerations

7.1. Questionnaire(s)

This project involves the use of voluntary and anonymous questionnaires to gather feedback from users. The purpose of the questionnaires is to inform the design, content, and functionality of the tools developed in this project.

- **Voluntary Participation:** All participants engage on a completely voluntary basis, with the option to withdraw at any time.
- **Anonymity and Privacy:** No personally identifiable information will be collected. Responses are anonymous and cannot be traced back to individual participants.
- **Data Usage:** Data will be analysed in aggregate form only and used exclusively for research and development within the scope of this project.

7.2. Web-Based Tool

The web application developed as part of this project adheres to strong ethical principles regarding user privacy and data protection.

- **No Personal Data Collection:** The application does not collect, process, or store any personal data.
- **No Tracking Technologies:** The website does not use cookies, analytics scripts, or third-party tracking services.
- **Local Storage Only:** All information related to the use of the tool is stored locally in the user's browser. This data remains entirely on the user's device and is never transmitted to any server.
- **User Autonomy and Transparency:** The design prioritizes user control, transparency, and minimal data exposure to ensure a secure and privacy-respecting experience.

By implementing these safeguards, the project maintains a high standard of ethical responsibility and respects the rights and privacy of all users.

8. ADL extraction

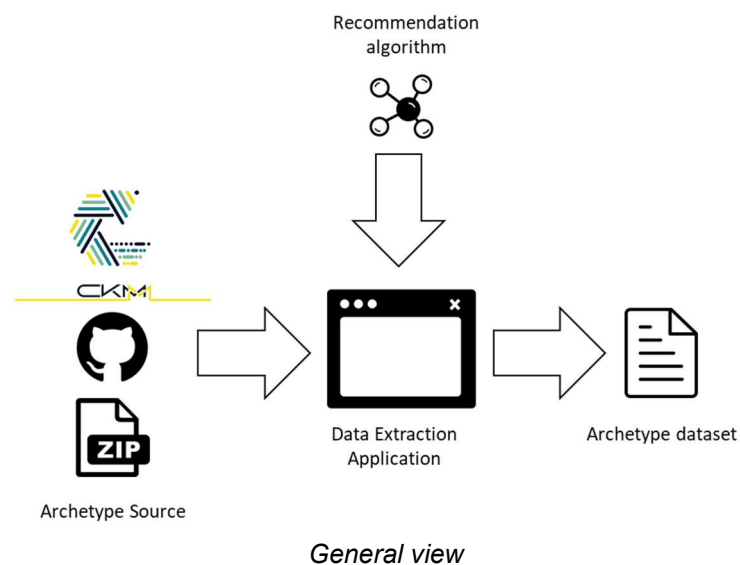
8.1. Introduction

For the final search, visualization, and planning tool, data from archetypes must be extracted. Hence, there is a need for an application that can execute this extraction periodically.

For this purpose, the **ADL extraction application** must fulfil the following functions:

In general, the application receives a collection of archetype files. The application extracts the files and parses them for the required information. Once all archetypes have been parsed, more information can be added - for example, by a recommendation algorithm that relates different archetypes to each other.

Once all information is accumulated, the application generates an archetype dataset file that can then be used by the visualization tool.



The functionalities can be specified further:

- **Data source:** The data source for extraction must be vendor-neutral and should not depend on a particular platform. The most sensible approach is to allow extraction from any collection of archetype files in a ZIP file. This typically includes:
 - **CKM:** CKM allows exporting archetypes in bulk as a ZIP file.
 - **GitHub:** Archetype collections can be downloaded from a GitHub repository as a ZIP file. Most CKMs have an associated GitHub repository.
 - **Local:** A custom collection of archetypes that a user can select from a local terminal.
- **ADL parsing:** The application must be capable of parsing ADL 1.4-compliant archetype files in bulk.
- **Data extraction:** Data extraction from the parsed archetype files must be limited to elements defined further on. In this case, only information from the English version of the archetype will be extracted. If the archetype is not available in English, the original language will be used.

- **Recommendation:** A basic recommendation should be provided to link similar or related archetypes. Linking will be based on semantics (for example, using the keywords of each archetype). The recommendation will be hard-coded into the resulting dataset. In the future, a more dynamic recommendation might be implemented, where the visualization tool adds recommendations based on user input.
- **GUI:** The application needs a GUI so a user can create the dataset manually whenever needed.
- **Automation:** In addition to manual execution, the application should have a headless mode that does not require user input. In this mode, the application should be executed automatically to update the dataset periodically. The execution does not have to be triggered by the application itself, but can be managed by external services.

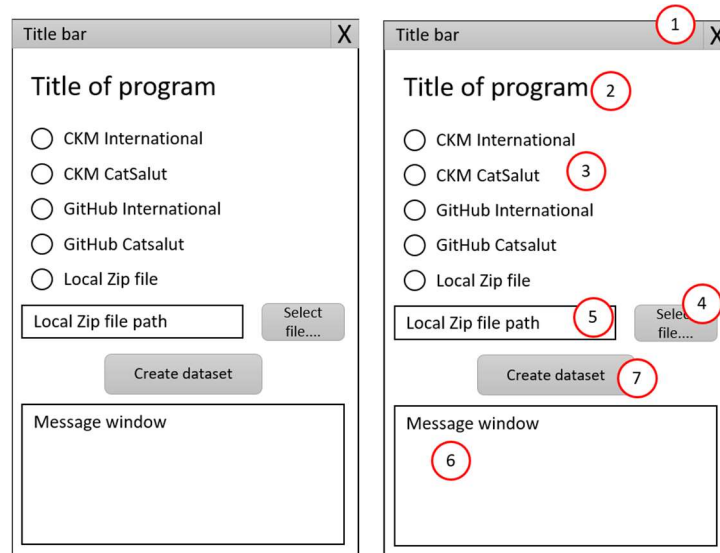
The planned development will take place predominantly in **Python code**. As a result, the python code will be provided as well as a compiled **Windows compatible executable** file.

8.2. ADL extraction GUI

From the previous general description of the application, the GUI must fulfil the following functions:

- Provide the possibility to select between different data sources.
- Give control over the generation of the dataset.
- Provide feedback during the entire process.

Based on these functionalities, a straightforward design can be drafted:



GUI wireframe

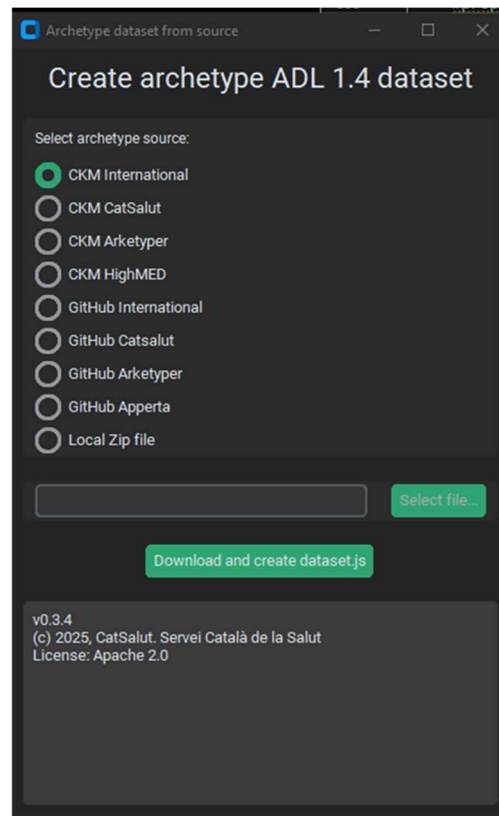
The elements are as follows:

1. **Title bar:** System title bar with common functionalities: "close", "minimize", and "maximize".
2. **Title label:** A label indicating the name of the application.
3. **Source selection:** Radio buttons that allow selection of one of the data sources. Planned options are:
 - International CKM
 - CKM CatSalut
 - Arketyper (Norwegian CKM)
 - CKM Highmed
 - GitHub of International CKM
 - GitHub of CKM CatSalut
 - GitHub of Arketyper
 - GitHub of the former Apperta CKM (UK)
 - Local ZIP file
4. **Select file path button:** If "Local ZIP file" is selected, this button is enabled. When pressed, the standard file selection window will open.
5. **File path textbox:** After selecting the local ZIP file, the path is shown here. Alternatively, a file path can be pasted here.
6. **Message box:** Text box that serves as output. On startup, it shows version and copyright information. When the process starts, feedback is given at every stage.
7. **Create dataset button:** This button is enabled when a data source is selected. Once pressed, it executes the process of download, extraction, and transformation.

Note that no selection for the output file folder is planned by now. **The output file is created and saved in the same directory as the application.**

Also note that no custom GitHub or CKM URLs are planned. This might be added in future versions.

The final resulting GUI was implemented in python using the *tkinter* and *customtkinter* libraries.



Final GUI

For **headless execution** of the program, set the *headless* variable to *True* and define in the variables *URL*, *CKMorGitHub*, *provenance* the data source:

```
[...]
# Set headless to True if you do not need the GUI (or False if you want to use the GUI)
headless = True
# Variables for running script headless (without GUI)
URL =
'https://ckm.openehr.org/ckm/retrieveResources?resourceType=archetype&format=ADL&state1=INITIAL&state2=DRAFT&state3=TEAMREVIEW&state4=REVIEWSUSPENDED&state5=PUBLISHED&state6=REASSESS_DRAFT&state7=REASSESS_TEAMREVIEW&state8=REASSESS_REVIEWSUSPENDED'
zipFileName = 'TempZipFile.zip'
CKMorGitHub = 'CKM' # "GITHUB" or "CKM", depending on source
provenance = 'CKM International'
[...]
```

8.3. Data Source

One of the main functionalities of the application is the possibility to download archetype collection in a ZIP file format from the CKMs or from GitHub repositories.

To get direct access to these zip files, the corresponding URLs have to be used.

GitHub repositories

Direct URLs to the GitHub repositories:

Source	URL
GitHub International	https://github.com/openEHR/CKM-mirror/archive/refs/heads/master.zip
GitHub CatSalut	https://github.com/CKMCatSalut/CKM-mirror/archive/refs/heads/master.zip
GitHub Arketyper	https://github.com/Arketyper-no/ckm/archive/refs/heads/master.zip
GitHub Apperta	https://github.com/AppertaFoundation/apperta-uk-ckm-mirror/archive/refs/heads/master.zip

CKM repositories

The direct URLs were obtained from the CKM's bulk export feature. They will perform a bulk download using the following settings:

The screenshot shows the 'Bulk Export Archetypes as ZIP' interface. At the top, there are tabs for 'Dashboard', 'Find Resources', and 'Bulk Export'. The main section is titled 'Bulk Export Archetypes as ZIP'. It contains several configuration options: 'Project / incubator' is set to 'All projects/incubators'; 'Reference model classes:' is set to 'All Classes'; 'Created on or after:' and 'Modified on or after:' are empty text boxes. Below these are checkboxes for 'States', 'Initial / Predraft', 'Draft', 'Team review', 'Review suspended', 'Published', 'Reassess (Draft)', 'Reassess (Team review)', 'Reassess (Review suspended)', 'Rejected', and 'Deprecated'. The 'Revision' section has two radio buttons: 'Get latest trunk revision' (selected) and 'Get latest published revision'. The 'Format' section has two radio buttons: 'ADL' (selected) and 'XML'. At the bottom, there is a 'Bulk Export' button.

State	Selected
States	☐
Initial / Predraft	☒
Draft	☒
Team review	☒
Review suspended	☒
Published	☒
Reassess (Draft)	☒
Reassess (Team review)	☒
Reassess (Review suspended)	☒
Rejected	☐
Deprecated	☐

REVISION

☒ Get latest trunk revision *i*

☐ Get latest published revision *i*

FORMAT

☒ ADL

☐ XML

Bulk Export

Equivalent CKM bulk export settings

For this project, it is preferable to have all publically available archetypes, including those coming from incubators.

Direct URLs to the CKM repositories:

Source	URL
CKM International	https://ckm.openehr.org/ckm/retrieveResources?resourceType=archetype&format=ADL&state1=INITIAL&state2=DRAFT&state3=TEAMREVIEW&state4=REVIEWSUSPENDED&state5=PUBLISHED&state6=REASSESS_DRAFT&state7=REASSESS_TEAMREVIEW&state8=REASSESS_REVIEWSUSPENDED
CKM CatSalut	https://ckm.salut.gencat.cat/ckm/retrieveResources?resourceType=archetype&format=ADL&state1=INITIAL&state2=DRAFT&state3=TEAMREVIEW&state4=REVIEWSUSPENDED&state5=PUBLISHED&state6=REASSESS_DRAFT&state7=REASSESS_TEAMREVIEW&state8=REASSESS_REVIEWSUSPENDED
CKM Highmed	https://ckm.highmed.org/ckm/retrieveResources?resourceType=archetype&format=ADL&state1=INITIAL&state2=DRAFT&state3=TEAMREVIEW&state4=REVIEWSUSPENDED&state5=PUBLISHED&state6=REASSESS_DRAFT&state7=REASSESS_TEAMREVIEW&state8=REASSESS_REVIEWSUSPENDED
CKM Arketyper	https://arketyper.no/ckm/retrieveResources?resourceType=archetype&format=ADL&state1=INITIAL&state2=DRAFT&state3=TEAMREVIEW&state4=REVIEWSUSPENDED&state5=PUBLISHED&state6=REASSESS_DRAFT&state7=REASSESS_TEAMREVIEW&state8=REASSESS_REVIEWSUSPENDED

Please note that the GitHub mirrors of CKMs exclusively contain archetypes from projects, excluding those from incubators. Conversely, the URLs for the CKM lead to zip files that encompass all archetypes from both incubators and projects.

If there is a need to download CKM data without incubators, the URLs should follow this format:

https://ckm.openehr.org/ckm/retrieveResources?resourceType=archetype&projectType=project&format=ADL&state1=INITIAL&state2=DRAFT&state3=TEAMREVIEW&state4=REVIEWSUSPENDED&state5=PUBLISHED&state6=REASSESS_DRAFT&state7=REASSESS_TEAMREVIEW&state8=REASSESS_REVIEWSUSPENDED

8.4. Data Elements

Definition of Data to be extracted

For the following definition of treatment of the ADL1.4 archetype files, it is important to determine the data elements that should be extracted from the files. The following key elements have been selected:

- **ID:** The archetype ID, such as “openEHR-EHR-ACTION.screening.v0”.
- **Class:** Identified in the archetype ID. For instance, “openEHR-EHR-ACTION.screening.v0” belongs to the class “ACTION”.
- **Publication state:** Represented in the ADL “description” section as: lifecycle_state = <“in_development”>.
- **License:** Located in the “description” section under “other_details”: [“licence”] = <“Creative Commons Attribution-ShareAlike 4.0
<<https://creativecommons.org/licenses/by-sa/4.0/>>”>
- **Revision:** Located in the “description” section under “other_details”: [“revision”] = <“0.0.1-alpha”>
- **Original language:** The original language is specified in the “language” section: original_language = <[ISO_639-1::en]>.
- **Language translations:** List of All Available Languages in the Archetype. Translation languages are listed in the “language” section under “translations”.
- **Date:** Authoring date, located in the “description” section under “original_author/date”.
- **Author name:** Original author name, located in the “description” section under “original_author/name”.
- **Author organisation:** Original author organisation, located in the “description” section under “original_author/organisation”.
- **Author email:** Original author email, located in the “description” section under “original_author/email”.
- **Purpose:** Located in the “description” section under “details/{lang}/purpose”.*
- **Use:** Located in the “description” section under “details/{lang}/use”.*
- **Misuse:** Located in the “description” section under “details/{lang}/misuse”.*
- **Copyright:** Located in the “description” section under “other_details” as: copyright = <“© openEHR Foundation”>
- **Inclusion and Exclusion statements:**
 - **Inclusion statements:** Inclusion of other archetypes, primarily Clusters. Inclusions are specified in the “definition” section of the model.
 - Example:
include
archetype_id/value matches {/openEHR-EHR-CLUSTER\anatomical_location(-[a-zA-Z0-9_]+)*\.v1|openEHR-EHR-CLUSTER\anatomical_location_circle(-[a-zA-Z0-9_]+)*\.v1|openEHR-EHR-CLUSTER\anatomical_location_relative(-[a-zA-Z0-9_]+)*\.v2/}
 - If the inclusion criteria encompass all archetypes (archetype_id/value matches {/.*/}), rather than listing each archetype, a single item should indicate that “any archetype” is linked.

- **Exclusion statements:** Exclusion of other archetypes, primarily Clusters. Exclusions are specified in the “definition” section.
 - **Example:**
exclude
archetype_id/value matches {/openEHR-EHR-CLUSTER\device(-[a-zA-Z0-9_+)*\v1/}
 - If the exclusion criteria encompass all archetypes (archetype_id/value matches {/.*/}), rather than listing each archetype, a single item should indicate that “any archetype” is linked.
- Exclusion and inclusion statements have an object for each occurrence in the resulting data set. Each object has the following attributes:
 - **code:** atnnnn code of the element that has the inclusion/exclusion statement.
 - **label:** The label of the element.
 - **type:** The type of the element.
 - **archetypes:** A list with all archetypes that fit the regex pattern provided in the inclusion/exclusion statement.
- **Concept name:** Found in the “ontology” section under “term_definitions/{lang}/items/at0000”.* In some cases might be "at0000.1"
- **Concept description:** Found in the “ontology” section under “term_definitions/{lang}/items/at0000”.* In some cases might be "at0000.1"
- **Items:** List of all data elements with at-code. Every item has attributes:
 - **code:** at-code of the element.
 - **label:** The label of the element.
 - **type:** The type of the element.
 - **description:** The description of the element.
 - **occurrence:** The occurrence of the element.
- **Keywords:** Keywords located in the “description” section under “details/{lang}/keywords”.*
- **Parent:** If the current archetype is a specialization, the parent archetype is found in the "specialise" section.
- **Children:** List of archetypes that have the current archetype as parent.
- **Structure:** A HTML representation of the “definition” section. Labels have been added to the at-codes.

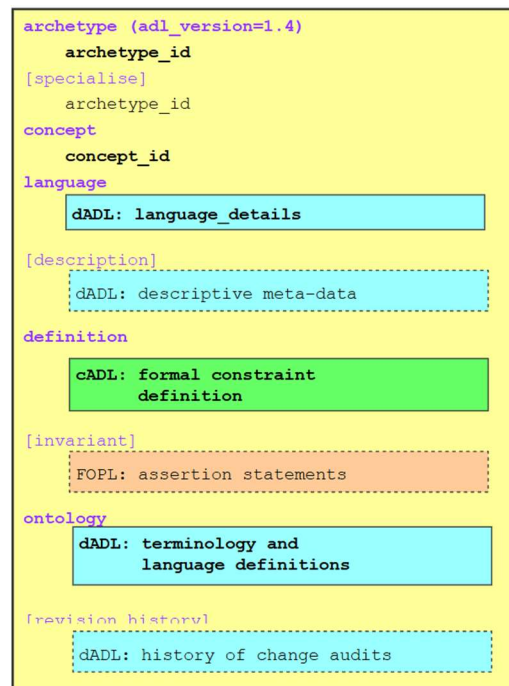
*The language ({lang}) should be English (“en”). If the archetype is not available in English, the keywords from the **original language** will be used.

Other information might be saved with every archetype, if found necessary during the development.

8.5. Parsing ADL Files

A crucial part of the application is parsing ADL 1.4 archetype files. Essentially, the application must open an archetype file and read it in a way that allows querying for specific data elements.

Parsing ADL .4 files is challenging because a single file can use multiple syntaxes. While the overall structure is ODIN-compliant (<https://odin-lang.org/>), ADL files also incorporate other syntaxes: cADL, dADL, and first-order predicate logic (FOPL).



The ADL1.4 archetype syntax

For more details, see the official ADL 1.4 specification (<https://specifications.openehr.org/releases/AM/latest/ADL1.4.html>).

ANTLR4 approach

ANTLR4 (ANother Tool for Language Recognition, <https://www.antlr.org/>) is a parser generator used to read, process, and translate structured text or binary files. It is widely adopted for building languages, tools, and frameworks.

The official openEHR GitHub provides two repositories related to ANTLR4 and ADL:

- openEHR-antlr4 (<https://github.com/openEHR/openEHR-antlr4>)
- adl-antlr (<https://github.com/openEHR/adl-antlr>)

However, it is unclear which project is best suited for ADL1.4. Notably, the *adl-antlr* repository states: "Currently this is limited to ADL2, but can be extended to ADL1.4 as needed."

After several attempts, the parsing with these tools could not be achieved with Python as of July 2025. Due to limited development time, it was not feasible to troubleshoot these issues or for the developer to become proficient in yet another programming language at this stage.

It was decided to pause efforts to use ANTLR4 for this project. However, it would be interesting to revisit this approach in the future, as leveraging official grammar files could provide a robust solution for ADL parsing.

Regex approach

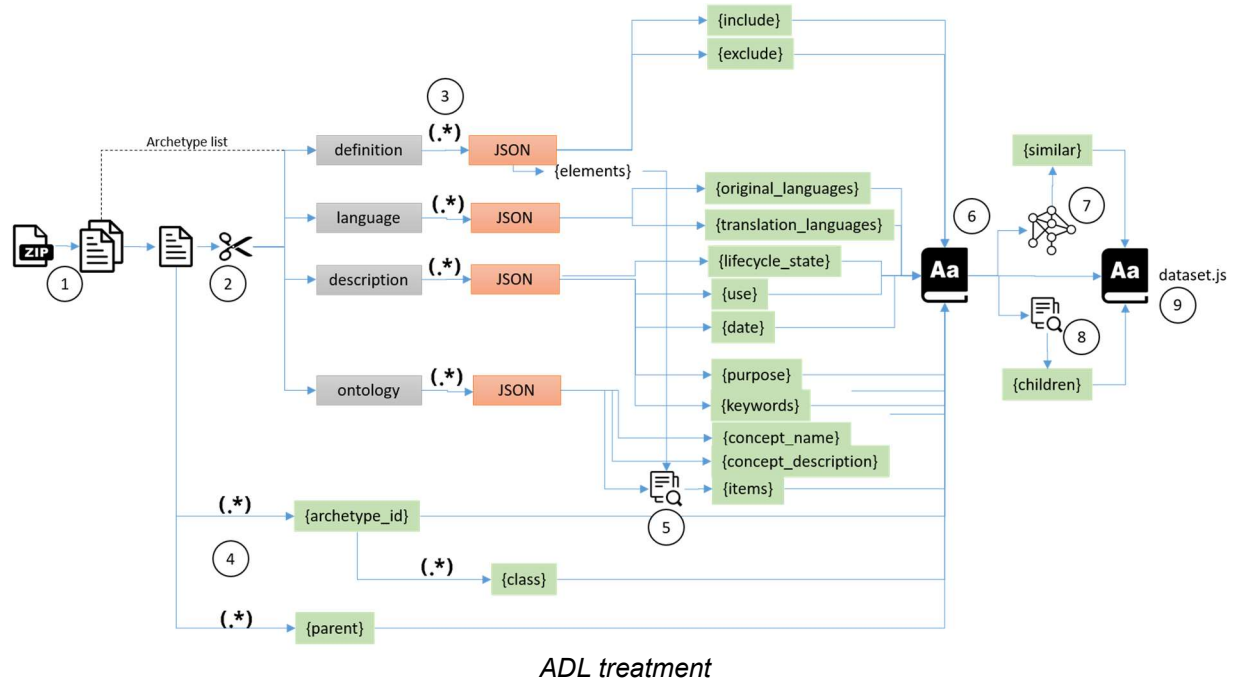
After looking into ANTLR4, the decision was made to parse the ADL files with a series of Regular Expressions (or Regex for short).

Regular expressions (regex) are sequences of characters that define search patterns, commonly used for matching, searching, and manipulating text. They are powerful tools for tasks such as data validation, extraction, and transformation. Online tools such as <https://regex101.com/> have been very helpful for this.

One problem of parsing of ADL files is that they consist of sections that use different syntaxes. In a general sense, the idea here is to proceed as follows:

1. Separate the different sections
2. Using Regex, transform the section text into a JSON compliant text
3. Convert text to JSON dictionary
4. Parse the JSON dictionary

The following diagram elaborates more detailed on the structures and the information gathered from them:



1. Extract ADL files from zip file
2. Separate every ADL file into sections
3. Transform every section into a parsable JSON format
4. Lookup at codes in the ontology section
5. Parse the original ADL file for archetype_id, class and parent information
6. Create a dictionary of all ADL file information gathered
7. Using keywords and item information, create a list of similar archetypes
8. Using parent information, create a list of children for all parent archetypes
9. Create the final data set

The developed parser uses regular expressions (regex) to detect semantic patterns in ADL files, extract information, and convert them into JSON-compatible semantics.

Archetype Section

The archetype data is extracted from the first section of the source ADL file.

Function	Regex Pattern
Extract archetype ID	.*(openEHR-.+-.+\.v[0-9]+)
Extract archetype class from archetype ID	openEHR-.+-(.+)\.v[0-9]+

Specialise Section

The archetype parent id is extracted from the specialise section of the source ADL file.

Function	Regex Pattern
Extract parent archetype ID	specialise[\t\n]+(openEHR-.+-.+\.v[0-9]+)

Pre-Processing for JSON sections

Before converting the definition, description, ontology and language section to a JSON structure, some changes have to be made.

Item/List	Regex Pattern	Original ADL Code Example	Converted JSON Code Example
Get rid of comments per line	(.*([\s\t]+--\s.*))	-- I am a comment	

Ontology, Description, and Language Sections

Below are the regex patterns used to transform the ontology, description and language sections into JSON-compliant format, with examples.

Single Line Expressions

Single line patterns are used to search for and replace single line key-value pairs.

Item/ List	Regex Pattern	Original ADL Code Example	Converted JSON Code Example
List	<code>(([A-Za-z_]+\s=\s<("[^"]+" ".*"? ".*?")>\n)</code>	<code>keywords = <"certificate", "death", "MCCD"></code>	<code>"keywords": ["certificate", "death", "MCCD"]</code>
Single item	<code>(([t+])([A-Za-z_0-9]+\s=\s<"(.*)">\n)</code>	<code>copyright = <"© openEHR Foundation"></code>	<code>"copyright": "© openEHR Foundation"</code>
Single item (key in brackets)	<code>(\[("[A-Za-z_0-9.,@-]+")\s=\s<"(.*)">\n)</code>	<code>["organisation"] = <"Forvaltningssenter EPJ, Helse-Nord RHF"></code>	<code>"organisation": "Forvaltningssenter EPJ, Helse-Nord RHF"</code>
List ("..." at the end)	<code>(([A-Za-z_0-9]+\s=\s<"(.*)" s\.\\.\\.>\n)</code>	<code>other_contributors = <"Heather Leslie, Atomica Informatics, Australia", ...></code>	<code>"other_contributors": ["Heather Leslie, Atomica Informatics, Australia"]</code>
Single item (item in brackets)	<code>(([A-Za-z_0-9]+\s=\s<\[(".*")\]>\n)</code>	<code>language = <[ISO_639-1::de]></code>	<code>"language": "ISO_639-1::de"</code>
Single item (key may be in brackets, item in brackets)	<code>(\[("[A-Za-z_0-9.\[\\]-]+")\s=\s<\[(".*")\]>\n)</code>	<code>["items[at0001]"] = <[SNOMED-CT::59276001]></code>	<code>"/items[at0001]": "SNOMED-CT::59276001"</code>
Single item (specific for terminology)	<code>(\[("[A-Za-z_0-9.-]+")\s=\s<["^"](.*)">\n)</code>	<code>["ac0002"] = <terminology:SNOMED-CT/408733002?subset=Diabetic%20Retinopathy%20Study%20field></code>	<code>"ac0002": "terminology:SNOMED-CT/408733002?subset=Diabetic%20Retinopathy%20Study%20field"</code>
Single item (empty item)	<code>(([A-Za-z_0-9]+\s=\s<>\n)</code>	<code>other_contributors = <></code>	<code>"other_contributors": ""</code>

Multiple Lines

These patterns are used to search for and replace multiple line key-value pairs, where texts or lists are separated by line breaks.

Item/ List	Regex Pattern	Original ADL Code Example	Converted JSON Code Example
Empty	<code>(([A-Za-z_0-9]+)\s=\s<([\t\s\n]+>\n)</code>	<code>author = < ></code>	<code>"author": ""</code>
Single item	<code>(([A-Za-z_]+)\s=\s<("[\s\S]*?")>\n)</code>	<code>description = <"Kennzeichen das angibt, ob es notwendig ist das Geburtsdatum nachzuverfolgen, um ein genaueres Datum zu erhalten."></code>	<code>"description": "Kennzeichen das angibt, ob es notwendig ist das Geburtsdatum nachzuverfolgen, um ein genaueres Datum zu erhalten."</code>
Single item (key in brackets)	<code>(\[("[A-Za-z_0-9.-]+")\s=\s<("[\s\S]*?")>\n)</code>	<code>["references"] = <"https://fhir.hl7.org.uk/STU3/StructureDefinition/CareConnect-Patient-1 cited 30-Jul-2018. https://fhir.hl7.org.uk/STU3/StructureDefinition/CareConnect-Practitioner-1 cited 30-Jul-2018."></code>	<code>"references": "https://fhir.hl7.org.uk/STU3/StructureDefinition/CareConnect-Patient-1 cited 30-Jul-2018. https://fhir.hl7.org.uk/STU3/StructureDefinition/CareConnect-Practitioner-1 cited 30-Jul-2018."</code>

Nested Structure

For nested structure, the approach is to convert "from outside to inside", starting from big indent to small indent using *str(level)*.

Structure	Regex Pattern	Original ADL Code (partially changed)	Converted JSON Code Example
Structure (key in brackets, <i>str(level)</i> = indent level)	<code>(\t{'+str(level)+'})\[\"([A-zÀ-ú0-9-._\s]+)\"]\s=\s<(\n[\"S\s]+?[\"])\t{'+str(level)+'}>\n</code>	<code>[\"ko\"] = < \"language\": \"ISO_639-1::ko\" \"author\": { \"name\": \"Seung-Jong Yu\" \"organisation\": \"NOUSCO Co., Ltd.\" } \"accreditation\": \"Certified Board of Family Medicine in South Korea\" ></code>	<code>\"ko\": { \"language\": \"ISO_639-1::ko\" \"author\": { \"name\": \"Seung-Jong Yu\" \"organisation\": \"NOUSCO Co., Ltd.\" } \"accreditation\": \"Certified Board of Family Medicine in South Korea\" }</code>
Structure (<i>str(level)</i> = indent level)	<code>(\t{'+str(level)+'}){([a-zA-Z0-9-._\s]+)\s=\s<(\n[\"S\s]+?[\"])\t{'+str(level)+'}>\n</code>	<code>author = < \"name\": \"Seung-Jong Yu\" \"organisation\": \"NOUSCO Co., Ltd.\" ></code>	<code>\"author\": { \"name\": \"Seung-Jong Yu\" \"organisation\": \"NOUSCO Co., Ltd.\" }</code>

Definition Section

Below are the regex patterns used to transform the definition section into JSON-compliant format, with examples.

In the definition section, a problem arises from the incorporation of different syntaxes. You can see clearly that the C_DV_QUANTITY section here is different from the rest:

```
protocol matches {
  ITEM_TREE[at0020] matches { -- Tree
    items cardinality matches {0..*; unordered} matches {
      ELEMENT[at0081] occurrences matches {0..1} matches { -- Testing Distance
        value matches {
          C_DV_QUANTITY <
            property = <[openehr::122]>
            list = <
              ["1"] = <
                units = <["ft_i"]>
                magnitude = <|>=0.0|>
                precision = <|1|>
              >
              ["2"] = <
                units = <"m">
                magnitude = <|>=0.0|>
                precision = <|1|>
              >
              ["3"] = <
                units = <"cm">
                magnitude = <|>=0.0|>
                precision = <|1|>
              >
              ["4"] = <
                units = <["in_i"]>
                magnitude = <|>=0.0|>
                precision = <|1|>
              >
            >
          >
        }
      }
    }
  }
}
```

In the definition section, we can find a mix of syntaxes. Example from openEHR-EHR-OBSERVATION.visual_acuity.v0.

Pre-processing - Ordinal Values

Item/List	Regex Pattern	Original ADL Code Example	Converted JSON Code Example
Find and handle ordinal value lists, example: 0.0 [local::at0030], 1.0 [local::at0031], 2.0 [local::at0032], 3.0 [local::at0033]	<code>(([-]*[0-9.]*)\\(\\[.*\\])[:,;]*)</code>	0 [local::at0004],	"0": "[local::at0004]"
Find and eliminate default ordinal value (single number in line)	<code>([0-9.]+?.\$)</code>	0	

Pre-processing - C_DV_QUANTITY sections**Single Line**

Item/List	Regex Pattern	Original ADL Code Example	Converted JSON Code Example
key = <"value">	<code>(([A-Z_a-z]+)\\s=\\s<"(.*)">)</code>	units = <"cm[H2O]">	"units": "cm[H2O]"
key = < value >	<code>(([A-Z_a-z]+)\\s=\\s< (.*) >\\n)</code>	precision = < 0 >	"precision": " 0 "
key = [value]	<code>(([A-Z_a-z]+)\\s=\\s< (.*) >)</code>	property = <[openehr::122]>	"property": "openehr::122"
key = <value>	<code>(([A-Z_a-z]+)\\s=\\s<(.*)>)</code>	precision = <1>	"precision": "1"

Multiple Lines

Item/List	Regex Pattern	Original ADL Code Example	Converted JSON Code Example
List elements	<code>(([\t]*)(. *))s=s<([\s\S]+?)>\n</code>	<code>["1"] = < "units": "cm[H2O]" "magnitude": " >=0.0 " "precision": " 1 " ></code>	<code>"1": { "units": "cm[H2O]" "magnitude": " >=0.0 " "precision": " 1 " }</code>
Lists	<code>(([\t]*)(. *))s=s<([\s\S]+?)>\n</code>	<code>list = < "1": { "units": "cm[H2O]" "magnitude": " >=0.0 " "precision": " 1 " } "2": { "units": "mm[Hg]" "magnitude": " >=0.0 " "precision": " 0 " } ></code>	<code>"list": { "1": { "units": "cm[H2O]" "magnitude": " >=0.0 " "precision": " 1 " } "2": { "units": "mm[Hg]" "magnitude": " >=0.0 " "precision": " 0 " } }</code>
Multiple line envelopes	<code>(([\t]*)(. *))s<([\s\S]+?)>\n</code>	<code>C_DV_QUANTITY < "property": "openehr::125" "list": { "1": { "units": "cm[H2O]" "magnitude": " >=0.0 " "precision": " 1 " } "2": { "units": "mm[Hg]" "magnitude": " >=0.0 " "precision": " 0 " } } ></code>	<code>"C_DV_QUANTITY": { "property": "openehr::125" "list": { "1": { "units": "cm[H2O]" "magnitude": " >=0.0 " "precision": " 1 " } "2": { "units": "mm[Hg]" "magnitude": " >=0.0 " "precision": " 0 " } } }</code>

Pre-processing - definition section

Item/List	Regex Pattern	Original ADL Code Example	Converted JSON Code Example
Get rid of space between ELEMENT and [atNNNN]	<code>(([A-Z_]+)\s(\[at.*\]))</code>	<code>CAPABILITY [at0000]</code>	<code>CAPABILITY[at0000]</code>
Delete cardinalities	<code>(cardinality\smatches\s\{.*?\})</code>	<code>cardinality matches {1..*; ordered}</code>	
Delete occurrences	<code>(occurrences\smatches\s\{.*?\})</code>	<code>occurrences matches {0..*}</code>	
Delete existences	<code>(existence\smatches\s\{.*?\})</code>	<code>existence matches {0..1}</code>	
Delete use_node lines	<code>(([\t]*)use_node.*)</code>	<code>use_node ELEMENT /data[at0001]/items[at0098]/items[at0099]</code>	

Nested Structure

Structure	Regex Pattern	Original ADL Code (partially changed)	Converted JSON Code Example
Adjust include lines	<code>((\t{'+str(level)+'})include\sarchetype_id/value\smatches\n)</code>	<code>include archetype_id/value matches</code>	<code>include archetype_id/value matches</code>
Adjust exclude lines	<code>((\t{'+str(level)+'})exclude\sarchetype_id/value\smatches\n)</code>	<code>exclude archetype_id/value matches</code>	<code>exclude archetype_id/value matches</code>
Add * to empty values	<code>(([\t]++){'+str(level)+'})\{ \n\t{'+str(level)+'}\}</code>	<code>{ }</code>	<code>{ "dummy_value": "" }</code>
Contract structures with only one value	<code>(([\t]*)([A-Za-z_0-9.-<>[\]]+))\smatches\n\t{'+str(level)+'}\{ \n\t{'+str(level)+'}\{ \n\t{'+str(level)+'}\{ \n\t{'+str(level)+'}\}</code>	<code>defining_code matches { [ac0000] }</code>	<code>"defining_code": "[ac0000]"</code>
Convert structures	<code>(([\t]*)([A-Za-z_0-9.-<>[\]]+))\smatches\n\t{'+str(level)+'}\{ \n([\s\S]+?)\n\t{'+str(level)+'}\}</code>	<code>DV_CODED_TEXT matches { "defining_code": "[ac0002]" }</code>	<code>"DV_CODED_TEXT": { "defining_code": "[ac0002]" }</code>
Replace include statements	<code>(([\t]+)include\n\t{'+str(level)+'}[\s\S]+?)(\})\{exclude)</code>	<code>include "archetype_id/value": "/openEHR-EHR-ITEM_TREE\om_tvangs middelvedtak(-[a-zA-Z0-9_+)*\v1/"</code>	<code>"include": { "archetype_id/value": "/openEHR-EHR-ITEM_TREE\om_tvangs middelvedtak(-[a-zA-Z0-9_+)*\v1/" }</code>
Replace exclude statements	<code>(([\t]+exclude\n\t{'+str(level)+'}[\s\S]+?)(\})\{include)</code>	<code>exclude "archetype_id/value": "/.*/"</code>	<code>"exclude": { "archetype_id/value": "/.*/" }</code>

Post-processing for JSON sections

For post processing the JSON structures of definition, description, ontology and language section, mainly commas have to be added to the resulting text created.

Function	Regex Pattern	Original ADL Code (mostly changed)	Converted JSON Code Example
Add commas between	<code>((".*?")([\n\t\s]+)(".*?"))</code>	<code>"original_language": "ISO_639-1::en"</code>	<code>"original_language": "ISO_639-1::en",</code>

items		"translations" "language": "ISO_639-1::ko" "author"	"translations", "language": "ISO_639-1::ko" "author"
Add commas between structures	(({})([\\n\\t\\s]+)(".*?"))	} "details"	}, "details"
Add commas between lists and others	((\\[\\])([\\n\\t\\s]+)(".*?"))	] "misuse"	], "misuse"

Testing of the parser

The parsing has been tested with a total of **1895** archetypes that have been downloaded from all available CKMs (International, HighMed, CatSalut, Arketyper, Alberta) and the Apperta GitHub (as of July, 2nd 2025). The parser had no problem with the archetypes.

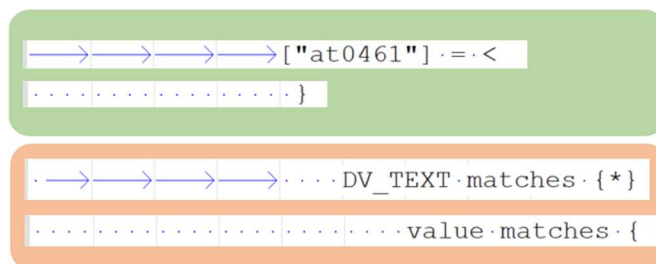
Periodically a full testing will be conducted to assure the function.

Incoherent archetype quality encountered during parsing

For parsing code, especially nested structures, it is very important that indentations (the spaces at the start of each line) are consistent. Normally, code will use either tabs or spaces for indentation. When spaces are used, they typically replace tabs, with 4 spaces being equivalent to one tab.

During the parsing process, it was found that not all archetypes adhere to this convention, which can create problems during parsing.

Specifically, some archetypes mix spaces and tabs in their indentations, while others use spaces, but not in a number divisible by 4. Both issues can lead to errors or unexpected behaviour when processing these files.



Indent formatting. Coherent tabs or spaces are standard. Mixing tabs and space number not divisible by 4 should be avoided.

In the following table, all **82** archetypes with problematic indentation are listed for reference:

Archetype	Mixed tabs and spaces	Spaces not divisible by 4
openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0	✓	✓
openEHR-DEMOGRAPHIC-ADDRESS.address.v0	✓	✓
openEHR-DEMOGRAPHIC-ADDRESS.electronic_communication.v0	✓	✓
openEHR-DEMOGRAPHIC-CAPABILITY.individual_credentials.v0	✓	✓
openEHR-DEMOGRAPHIC-CLUSTER.biometric_identifier_iso.v0	✓	✓
openEHR-DEMOGRAPHIC-CLUSTER.individual_credentials_iso.v0	✓	✓
openEHR-DEMOGRAPHIC-CLUSTER.individual_provider_credentials_iso.v0	✓	✓
openEHR-DEMOGRAPHIC-CLUSTER.person_additional_data_br.v0	✗	✓
openEHR-DEMOGRAPHIC-CLUSTER.person_additional_data_iso.v0	✓	✓
openEHR-DEMOGRAPHIC-CLUSTER.person_birth_data_iso.v0	✗	✓
openEHR-DEMOGRAPHIC-CLUSTER.person_death_data_iso.v0	✓	✗
openEHR-DEMOGRAPHIC-CLUSTER.person_identifier-provider.v0	✓	✓
openEHR-DEMOGRAPHIC-CLUSTER.person_other_birth_data_br.v0	✓	✓
openEHR-DEMOGRAPHIC-CLUSTER.provider_identifier.v0	✓	✓
openEHR-DEMOGRAPHIC-ITEM_TREE.person_details.v0	✓	✓
openEHR-DEMOGRAPHIC-ORGANISATION.organisation.v0	✓	✗
openEHR-DEMOGRAPHIC-PARTY_IDENTITY.organisation_name.v0	✓	✓
openEHR-DEMOGRAPHIC-PARTY_IDENTITY.person_name.v0	✓	✓
openEHR-DEMOGRAPHIC-PERSON.person-patient.v0	✓	✓
openEHR-DEMOGRAPHIC-PERSON.person.v0	✓	✓
openEHR-DEMOGRAPHIC-ROLE.healthcare_consumer.v0	✓	✓
openEHR-DEMOGRAPHIC-ROLE.individual_provider.v0	✓	✓
openEHR-EHR-ACTION.medication.v1	✗	✓
openEHR-EHR-ADMIN_ENTRY.triage.v0	✗	✓
openEHR-EHR-CLUSTER.address.v1	✗	✓
openEHR-EHR-CLUSTER.auscultation_bowel.v0	✗	✓
openEHR-EHR-CLUSTER.device.v1	✗	✓
openEHR-EHR-CLUSTER.dwelling.v0	✗	✓
openEHR-EHR-CLUSTER.examination_anterior_chamber_eye.v0	✗	✓
openEHR-EHR-CLUSTER.examination_optic_disc.v0	✗	✓

Archetype	Mixed tabs and spaces	Spaces not divisible by 4
openEHR-EHR-CLUSTER.imaging_exam-hip_joint.v1	✗	✓
openEHR-EHR-CLUSTER.imaging_exam-spine.v0	✗	✓
openEHR-EHR-CLUSTER.inspired_oxygen.v1	✗	✓
openEHR-EHR-CLUSTER.laboratory_test_analyte.v1	✗	✓
openEHR-EHR-CLUSTER.language.v1	✗	✓
openEHR-EHR-CLUSTER.medication.v2	✗	✓
openEHR-EHR-CLUSTER.microscopy_breast_carcinoma.v1	✗	✓
openEHR-EHR-CLUSTER.microscopy_renal_biopsy_non_neoplastic.v0	✗	✓
openEHR-EHR-CLUSTER.problem_qualifier.v2	✗	✓
openEHR-EHR-COMPOSITION.encounter.v1	✗	✓
openEHR-EHR-EVALUATION.advance_care_directive.v2	✗	✓
openEHR-EHR-EVALUATION.clinical_synopsis.v1	✗	✓
openEHR-EHR-EVALUATION.communication_capability.v1	✗	✓
openEHR-EHR-EVALUATION.medication_summary.v1	✗	✓
openEHR-EHR-EVALUATION.occupation_summary.v1	✗	✓
openEHR-EHR-EVALUATION.reason_for_encounter.v1	✗	✓
openEHR-EHR-EVALUATION.social_network-JM.v0	✗	✓
openEHR-EHR-EVALUATION.social_network.v1	✗	✓
openEHR-EHR-EVALUATION.vaccination_summary.v0	✗	✓
openEHR-EHR-EVALUATION.vaping_summary.v0	✗	✓
openEHR-EHR-INSTRUCTION.medication_order.v3	✗	✓
openEHR-EHR-INSTRUCTION.request-report.v1	✗	✓
openEHR-EHR-OBSERVATION.atria_stroke_risk.v0	✗	✓
openEHR-EHR-OBSERVATION.blood_pressure.v2	✗	✓
openEHR-EHR-OBSERVATION.body_composition.v0	✗	✓
openEHR-EHR-OBSERVATION.body_mass_index.v2	✗	✓
openEHR-EHR-OBSERVATION.body_temperature.v2	✗	✓
openEHR-EHR-OBSERVATION.body_weight.v2	✗	✓
openEHR-EHR-OBSERVATION.braden_scale.v1	✗	✓
openEHR-EHR-OBSERVATION.braden_scale_q.v0	✗	✓
openEHR-EHR-OBSERVATION.eq_5d_5l.v0	✗	✓
openEHR-EHR-OBSERVATION.experiment.v0	✗	✓

Archetype	Mixed tabs and spaces	Spaces not divisible by 4
openEHR-EHR-OBSERVATION.has_bled.v0	✗	✓
openEHR-EHR-OBSERVATION.height.v2	✗	✓
openEHR-EHR-OBSERVATION.honos.v0	✗	✓
openEHR-EHR-OBSERVATION.howru.v1	✗	✓
openEHR-EHR-OBSERVATION.iciq_ui_short.v0	✗	✓
openEHR-EHR-OBSERVATION.imaging_exam_result.v1	✗	✓
openEHR-EHR-OBSERVATION.laboratory_test_result.v1	✗	✓
openEHR-EHR-OBSERVATION.medication_statement-JM.v0	✗	✓
openEHR-EHR-OBSERVATION.medication_statement.v0	✗	✓
openEHR-EHR-OBSERVATION.nihss.v0	✗	✓
openEHR-EHR-OBSERVATION.pregnancy_finding.v0	✗	✓
openEHR-EHR-OBSERVATION.progress_note.v1	✗	✓
openEHR-EHR-OBSERVATION.pulse.v2	✗	✓
openEHR-EHR-OBSERVATION.pulse_oximetry.v1	✗	✓
openEHR-EHR-OBSERVATION.respiration.v2	✗	✓
openEHR-EHR-OBSERVATION.sara_scale.v0	✗	✓
openEHR-EHR-OBSERVATION.substance_use.v0	✗	✓
openEHR-EHR-OBSERVATION.symptom_sign_screening.v1	✗	✓
openEHR-EHR-OBSERVATION.travel_screening.v0	✗	✓
openEHR-EHR-OBSERVATION.tympanogram_226hz.v0	✗	✓

Table data result from archetype extraction of the international CKM, 2nd of July 2025

8.6. Data Analysis

Data provided here has been gathered from archetypes extracted **02/07/2025** from the **international CKM**.

For this analysis, data from **917** publicly available archetypes in the international CKM were utilized. This includes all public accessible archetypes in incubators and projects.

Relationships: Parent-child, Inclusion and Exclusion

Understanding the relationships between archetypes is important for effective modelling and reuse within the openEHR ecosystem. Archetypes can be related through specialization (parent-child relationships), as well as through explicit inclusion or exclusion of other archetypes.

Parent-Child (Specialization) Relationships

The specialization of archetype provide a possible inheritance and extension of clinical models with the drawback of creating complex dependencies.

- Archetypes that act as **parents** (specialized): 32 (**3.5%**)
- Archetypes that are **children** (specializations): 128 (**14.0%**)

Inclusion and Exclusion Relationships

Inclusion relationships indicate which archetypes are explicitly referenced or incorporated within another archetype. *Exclusion* relationships, though less common, specify archetypes that should be explicitly omitted, helping to clarify boundaries and avoid ambiguity.

- Archetypes with **INCLUSION** specified: 669 (**73.0%**)
- Archetypes with **EXCLUSION** specified: 29 (**3.2%**)

Of the exclusion statements, all but one are excluding “all archetypes”, which is always paired with an inclusion statement, so the combination of both is “include these specified archetype, but exclude all others”.

There is only **one case** where a **specific** exclusion is defined:

In the archetype *openEHR-EHR-OBSERVATION.howru.v1*, specifically in the CLUSTER “Device” (at0038), there is a specific exclusion of ‘openEHR-EHR-CLUSTER.device.v1’ and its specialisations.

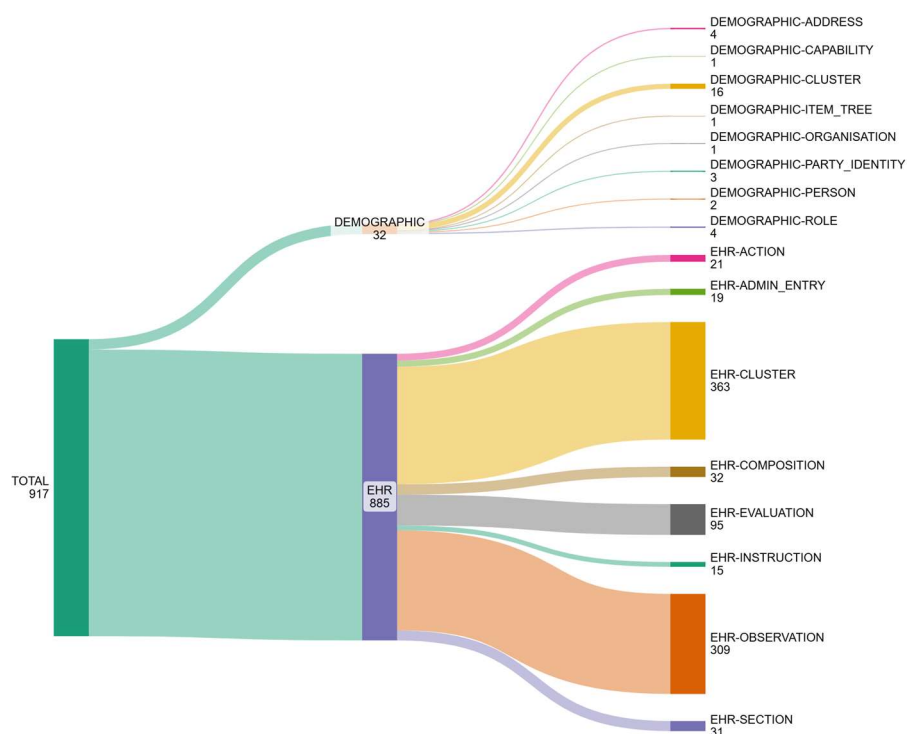
Archetype Classes

Archetype classes are fundamental for designing archetypes with their intended use in the clinical pathway in mind. In theory, all classes that inherit from the LOCATABLE class can be archetyped according to the openEHR Reference Model.

The classes of archetypes found in the analysed dataset are as follows:

- **EHR**
 - ACTION
 - ADMIN_ENTRY
 - CLUSTER
 - COMPOSITION
 - EVALUATION
 - INSTRUCTION
 - OBSERVATION
 - SECTION
- **DEMOGRAPHIC**
 - ADDRESS
 - CAPABILITY
 - CLUSTER
 - ITEM_TREE
 - ORGANISATION
 - PARTY_IDENTITY
 - PERSON
 - ROLE

The number of different archetype classes found is shown in this Sankey diagram:



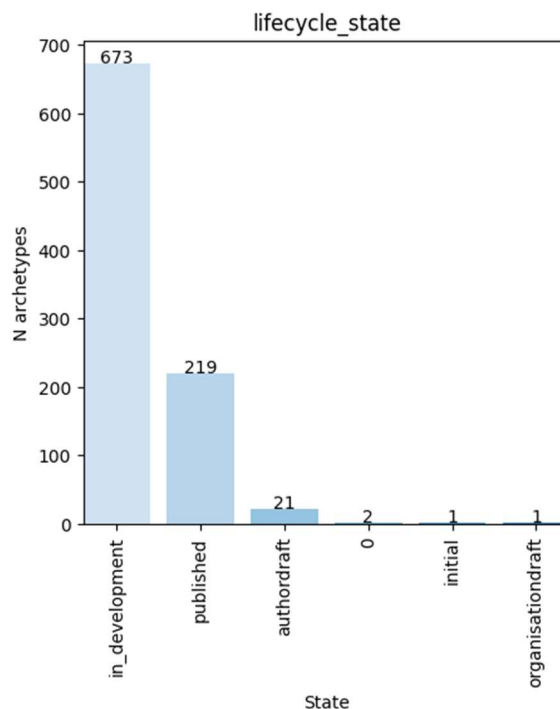
Lifecycle States

Lifecycle states are a critical aspect of openEHR archetype management, as they indicate the maturity, stability, and intended use of each archetype. Properly assigned lifecycle states help ensure clarity for developers, reviewers, and implementers and support the overall quality and interoperability of clinical models.

In the current analysis, **6** different lifecycle states were found among the archetypes:

- 0
- organisationdraft
- published
- authordraft
- initial
- in_development

In the following graph, the number of archetypes are shown for each lifecycle state:



According to the openEHR Resource Model specification (https://specifications.openehr.org/releases/BASE/development/resource.html#_resource_description_class), the official lifecycle states are: **unmanaged**, **in_development**, **release_candidate**, **published**, **rejected** and **deprecated**.

Outlier States Identified

The following states were found in the analysed archetypes but do not align with the official list of possible states:

lifecycle_state	archetype
0	openEHR-EHR-CLUSTER.person_name_isa.v1
0	openEHR-EHR-OBSERVATION.heart_failure_symptom_questionnaire.v1
authordraft	openEHR-EHR-ACTION.procedure-Hip_arthroplasty_previous_procedure_res.v1
authordraft	openEHR-EHR-ACTION.procedure-hip_arthroplasty_revision_previous_procedure_res.v1
authordraft	openEHR-EHR-ACTION.procedure-primary_hip_arthroplasty_res.v1
authordraft	openEHR-EHR-ACTION.procedure-revision_hip_arthroplasty_res.v1
authordraft	openEHR-EHR-CLUSTER.address_isa.v1
authordraft	openEHR-EHR-CLUSTER.death_details_parent.v1
authordraft	openEHR-EHR-CLUSTER.dermatology_therapy_summary_detail.v1
authordraft	openEHR-EHR-CLUSTER.diabetic_retinopathy_screening_result.v1
authordraft	openEHR-EHR-CLUSTER.exam_anterior_chamber.v1
authordraft	openEHR-EHR-CLUSTER.exam_lens.v1
authordraft	openEHR-EHR-CLUSTER.healthcare_professional_parent.v1
authordraft	openEHR-EHR-CLUSTER.healthcare_provider_parent.v1
authordraft	openEHR-EHR-CLUSTER.hip_procedure_valdoltra.v1
authordraft	openEHR-EHR-CLUSTER.person_identifiable_parent.v1
authordraft	openEHR-EHR-CLUSTER.photocoagulation_details.v1
authordraft	openEHR-EHR-CLUSTER.therapeutic_decision_dr.v1
authordraft	openEHR-EHR-EVALUATION.medication_safety_event.v1
authordraft	openEHR-EHR-EVALUATION.recommendation-DR_treatment.v1
authordraft	openEHR-EHR-INSTRUCTION.request-report.v1
authordraft	openEHR-EHR-OBSERVATION.dermatology_therapy_summary.v1
authordraft	openEHR-EHR-OBSERVATION.howru.v1
organisationdraft	openEHR-EHR-CLUSTER.exam_posterior_chamber.v1
initial	openEHR-EHR-CLUSTER.microscopy_breast_carcinoma.v1

It is highly recommended to review and address the **25** identified outlier states. Aligning all archetypes with the official lifecycle states will enhance the reliability, maintainability, and interoperability of the archetype repository, and ensure compliance with openEHR best practices.

Date of Authorship

All but **12** archetypes have a date of authoring. The following archetypes are the ones without date:

- openEHR-EHR-CLUSTER.anatomical_location_gingivae.v0
- openEHR-EHR-CLUSTER.anatomical_location_tooth.v0
- openEHR-EHR-CLUSTER.examination_dentition.v0
- openEHR-EHR-CLUSTER.examination_gingivae.v0
- openEHR-EHR-CLUSTER.examination_occlusal.v0
- openEHR-EHR-CLUSTER.examination_oral_membrane.v0
- openEHR-EHR-CLUSTER.examination_teeth.v0
- openEHR-EHR-CLUSTER.image_acquisition_details.v0
- openEHR-EHR-OBSERVATION.charlson_comorbidity_index.v2
- openEHR-EHR-OBSERVATION.examination_oral.v0
- openEHR-EHR-OBSERVATION.infant_feeding.v0
- openEHR-EHR-SECTION.result_details.v0

Date Format Consistency

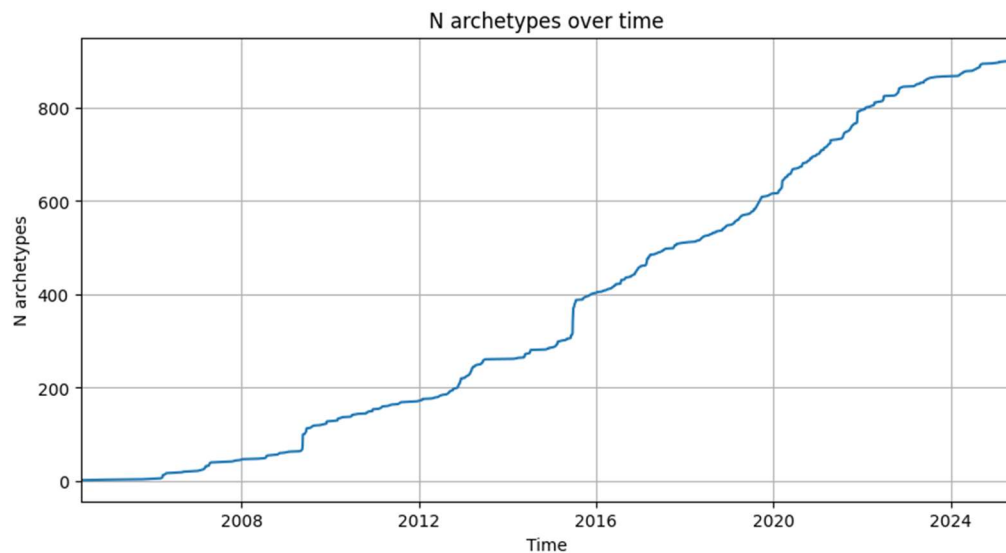
The majority of archetypes use the standard date format *YYYY-MM-DD*. However, **19 archetypes** were found to use non-standard or inconsistent date formats, which may cause issues when parsing or processing archetype data.

Archetype	Date Format Used
openEHR-EHR-ACTION.informed_consent.v0	09-10-2012
openEHR-EHR-CLUSTER.case_identification.v0	08/12/2009
openEHR-EHR-CLUSTER.macroscopy_colorectal_carcinoma.v0	17/06/2009
openEHR-EHR-CLUSTER.microscopy_colorectal_carcinoma.v0	17/06/2009
openEHR-EHR-CLUSTER.microscopy_lymphoma.v0	26/04/2009
openEHR-EHR-CLUSTER.microscopy_melanoma.v0	17/05/2009
openEHR-EHR-CLUSTER.microscopy_prostate_carcinoma.v0	17/06/2009
openEHR-EHR-CLUSTER.notifiable_condition.v0	08/12/2009
openEHR-EHR-CLUSTER.outbreak_identification.v0	08/12/2009
openEHR-EHR-CLUSTER.tumour_invasion.v0	27/07/2009
openEHR-EHR-CLUSTER.ventilator_settings2.v0	25/10/2010
openEHR-EHR-COMPOSITION.obstetric_history.v0	02-01-2012
openEHR-EHR-EVALUATION.contraindication-intravitreal_antiVEGF.v0	14-11-2012
openEHR-EHR-EVALUATION.contraindication.v1	14-11-2012
openEHR-EHR-EVALUATION.medication_safety_event.v1	25/06/2012
openEHR-EHR-OBSERVATION.howru.v1	03/03/2010
openEHR-EHR-OBSERVATION.speech.v0	10/11/2007
openEHR-EHR-OBSERVATION.tobacco_use.v0	6/5/2009
openEHR-EHR-SECTION.lab_test_report.v0	25/03/2010

The use of date and a consistent use of the `YYYY-MM-DD` format is recommended to ensure reliable parsing and interoperability of archetype metadata.

Archetypes over Time

Using the date information from the archetypes, the evolution of the number of archetypes over time can be visualized:



Fun Facts:

- **The oldest archetype:** *openEHR-EHR-OBSERVATION.apgar.v2* authored on **2004-05-18**.
- **The oldest archetype that is (still) not published:** *openEHR-EHR-OBSERVATION.modified_barthel_index.v0* authored on **2006-03-22**.

Keywords

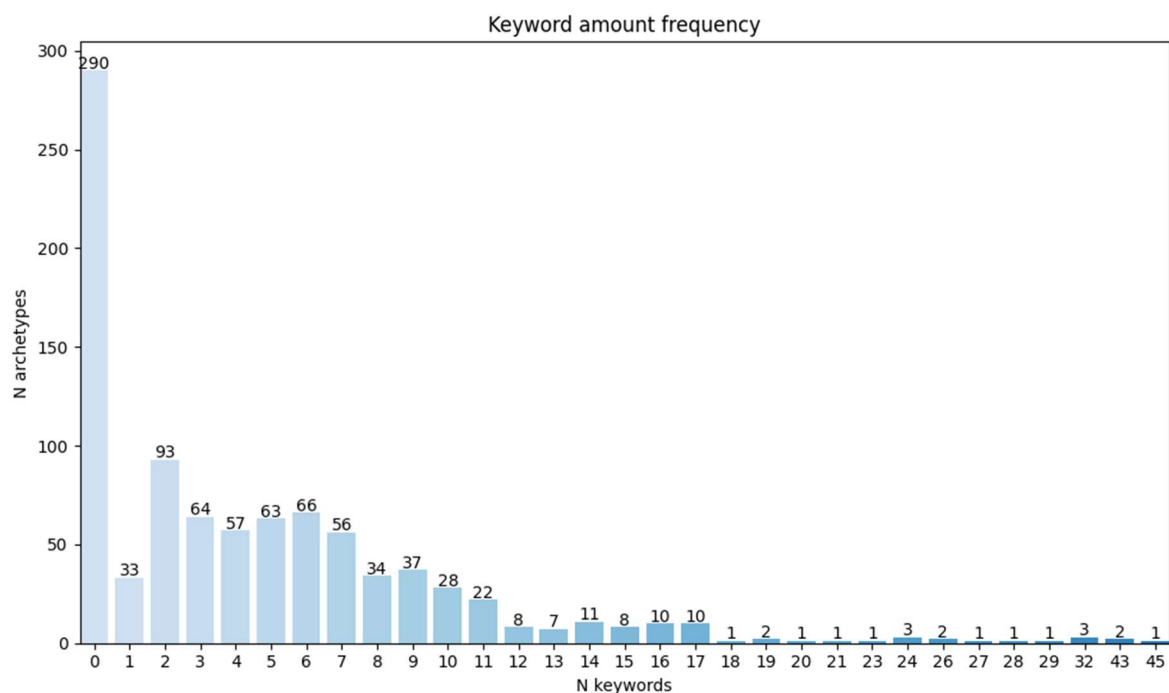
Keywords play a crucial role in the organization, discoverability, and semantic linking of openEHR archetypes. They help modellers and users quickly identify relevant archetypes and facilitate advanced features such as recommendation engines and semantic search. The following analysis provides insights into the usage and distribution of keywords across the archetype repository.

Keyword Usage Statistics

Note that all keywords were converted to lower case.

- **Total unique keywords used:** 2071
- **Archetypes without keywords:** 290 (31.6%)

The **absence of keywords** in nearly a third of archetypes can complicate semantic linking and reduce the effectiveness of search and recommendation algorithms. Ensuring that each archetype is assigned a reasonable set of keywords is therefore highly recommended.



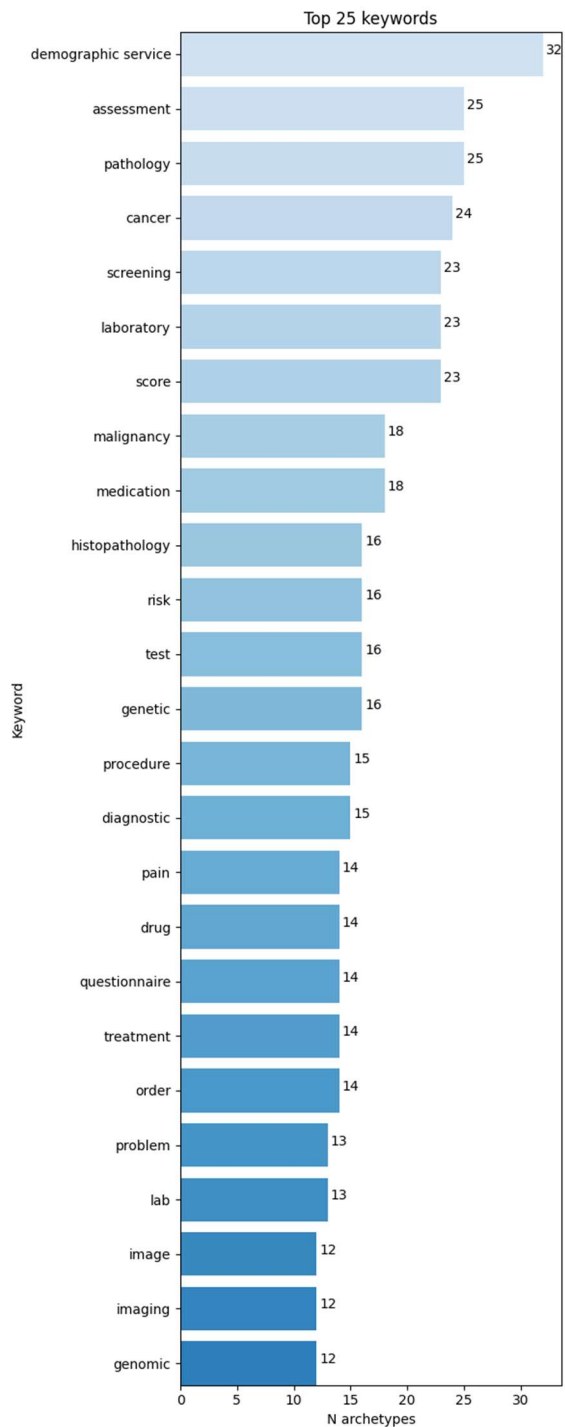
The average and median number of keywords per archetype can be calculated:

- **Mean** number of keywords per archetype: 4.6
- **Median:** 3

There is no strict rule for the number of keywords per archetype, but a range of **5 to 10 keywords** might be more reasonable. Assigning too few keywords may limit discoverability, while excessive use can reduce their effectiveness and introduce noise.

For example, the archetype with the most keywords is *openEHR-EHR-EVALUATION.smokeless_tobacco_summary.v1*, which has **45 keywords**.

To give an impression of the most used keywords, here is the ranking of the **top 25 keywords**:



8.7. Recommendation

This project aims to provide a tool that empowers users to search for archetypes that fit a list of data element labels, improving the discoverability and relevance of archetype suggestions.

Key terminology: Throughout this document, we distinguish between two groups of archetypes based on their similarity associations:

- **"0 similar"**: Archetypes that do *not* have any similar archetypes associated (i.e., no recommendations can be made for them).
- **"10 similar"**: Archetypes that have *10 or more* similar archetypes associated (i.e., they are highly connected in the recommendation system).

These terms will be used to describe the distribution and goals of the recommendation algorithm.

Searches can be conducted by archetype ID, keywords, items, and description texts. In practice, this often yields a list of archetypes containing the search term, which may result in too few or too many results. Archetypes with similar keywords or text items are not considered in this search. As a result, some archetypes are difficult to find. They might not contain the search term but may be related or similar in topic to the searched item.

To make these "obscure" archetypes accessible via search, this project intends to create a recommendation algorithm capable of suggesting similar or related archetypes.

The implementation of recommendations in this project should follow two general guidelines: recommendations should be **helpful** and **unintrusive**. The Microsoft Assistant is a good example of an intrusive recommendation system that was generally not liked by users (see Swartz, L. 2003).

When to Recommend

This project has been divided into two parts:

1. ADL data extraction
2. Archetype search, visualization, and planning

Logically, a recommendation for similar or related archetypes can be created in either part. The advantages and disadvantages are as follows:

Where?	Advantages	Disadvantages
ADL data extraction	• Control over the recommendation algorithm, based on a fixed data set. • Less computational effort in the visualization tool.	• Does not take user input into account.
Archetype search, visualization, and planning	• Adaptable to the current state of the user's project.	• More computational effort in the visualization tool. • Results are not foreseeable.

In this project, as a first step, recommendations will be created during the ADL extraction phase. Implementing patterns abstracted from modellers' previous experience and user input into the visualization tool would exceed the timeframe and resources of this project.

In the future, a hybrid approach might be considered, where a fixed set of recommendations is created in the ADL extraction phase, and in the subsequent visualization, the algorithm learns from the user's selection of data elements and archetypes to create more tailored recommendations.

How to Recommend

Recommender systems or recommendation engines help users discover new items of interest. Whether suggesting a movie, product, or song, these systems analyse available information to offer relevant and personalized options.

The most common types of recommendation systems are **content-based filtering**, **collaborative filtering**, and **hybrid approaches** (https://en.wikipedia.org/wiki/Recommender_system).

- **Content-based filtering** recommends items based on the attributes of items a user has previously interacted with, such as genre, actors, or keywords, without considering other users' preferences.
- **Collaborative filtering** makes recommendations by analysing the behaviour and preferences of many users, suggesting items that similar users have liked.
- **Hybrid recommendation systems** combine both content-based and collaborative methods to leverage the strengths of each and provide more accurate suggestions.

For this project, **content-based filtering** is considered most suitable, as it focuses on the attributes of items and does not rely on data from other users. Any other approach would need a whole infrastructure for managing user accounts, relating users with each other and create a rating system for archetypes. This would be out of the scope of this project.

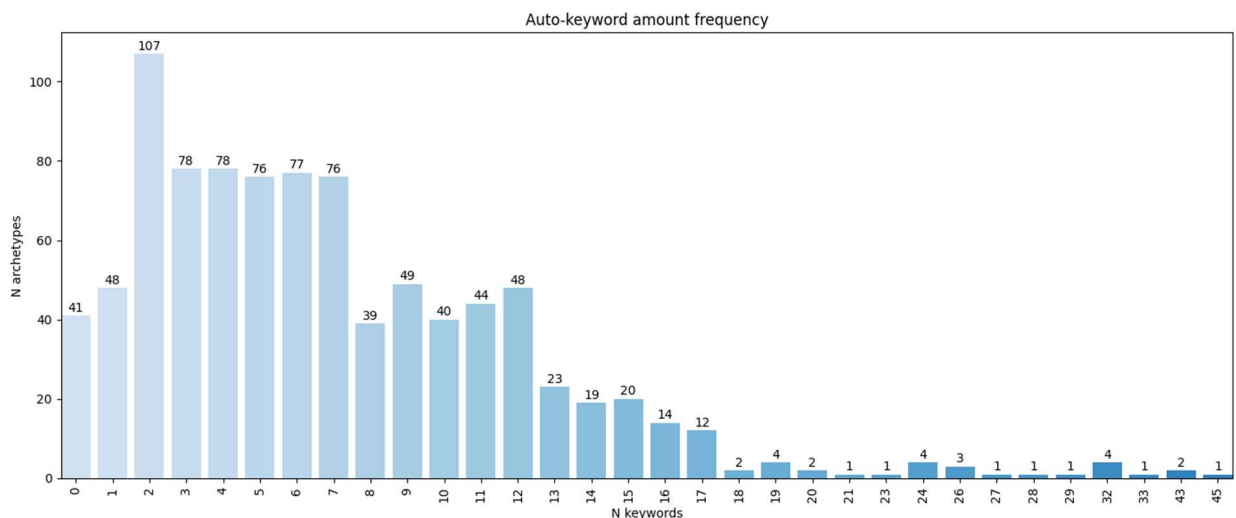
Adding content for content-based filtering

The "Auto-keyword" approach

As mentioned previously, the **lack of keywords in about on third of archetypes** can be a problem for the later generation of recommendations based on keywords.

To circumvent this, an additional attribute has been generated for the archetype dataset. With the list of >2000 keywords on hand, an algorithm was used to search for any of those keywords in the "purpose" and "concept description" for each archetype without keywords. If found, they were added to the "auto-keyword" attribute of this archetype. For archetypes with keywords, the keyword list was copied to the "auto-keyword" of the archetype.

This reduces the number of **keyword-less archetypes** to **41**. The distribution of number of archetypes with a certain number of "auto-keywords" is depicted in the following graph:

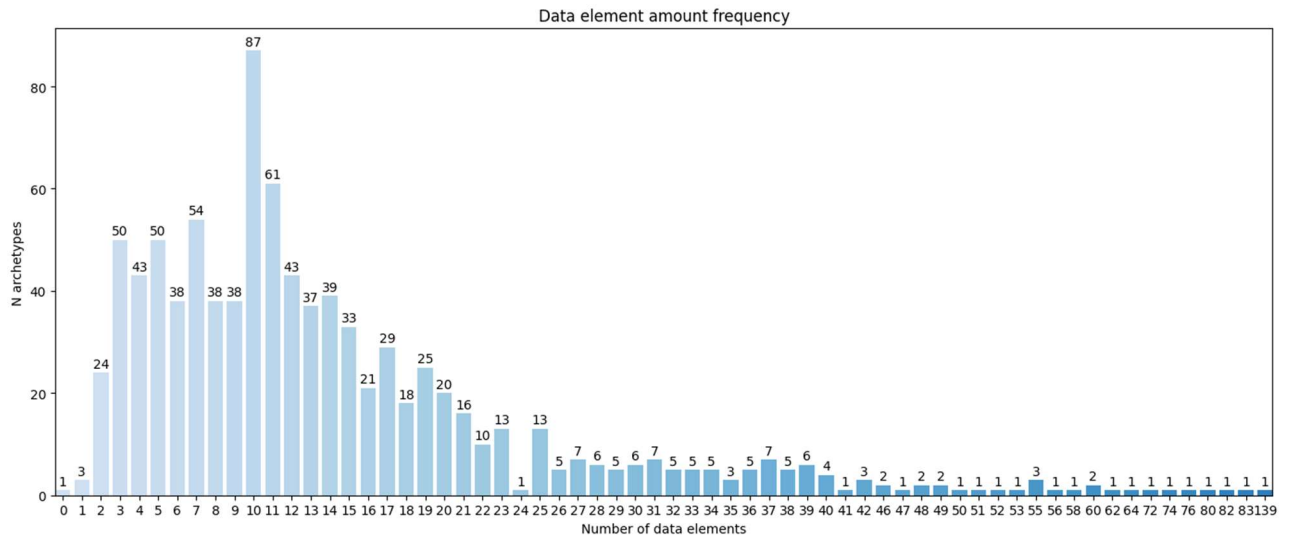


The average and median number of auto-keywords per archetype can be calculated:

- **Mean** number of auto-keywords per archetype: 7.0
- **Median**: 6

Data Element Labels

Because semantic linking of the archetypes by keywords might be limited, it was considered to also include the labels of data elements used.



The average and median number of data elements per archetype can be calculated:

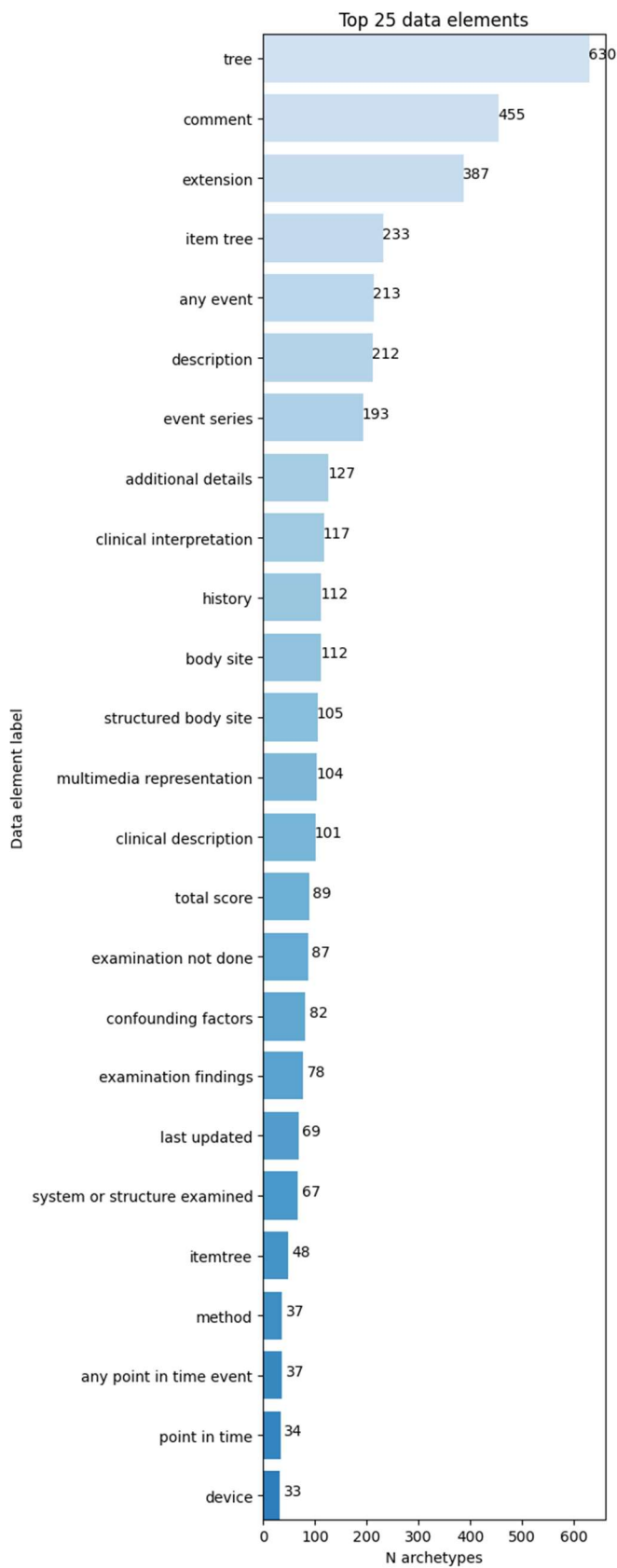
- **Mean** number of data elements per archetype: 14.4
- **Median**: 11

Fun facts:

- There are **6701** different element labels between all archetypes.
- The archetype with most items is openEHR-EHR-OBSERVATION.ygtss_revised.v1 with **139 data elements**.

To make the semantic linking effective, it will be important not to use labels that are used in the majority of archetypes. For example *tree* or *item tree* are standard labels in openEHR that do not add to the significance of the archetype itself.

Here the most common labels are shown that might be avoided for the recommendation engine.



Content-Based Filtering by Semantic Linking of Archetypes

Level 1 – Keywords

The first level of semantic linking applied here is the number of keywords that two archetypes have in common. To quantify the similarity between archetypes and create a "high score" list, a simple formula can be applied:

$$score = N_{kw}$$

Where:

- *score*: Total score for the similarity of the archetypes.
- N_{kw} : Number of identical keywords.

Here, the similarity score is equal to the number of common keywords.

As shown in the data analysis of extracted ADL data from the international CKM, about **a third of archetypes do not have keywords** ("0 similar" archetypes). This limits the application of this scoring system considerably and contributes to the group of "0 similar" archetypes.

Level 2 – Auto-keywords

To reduce the number of "0 similar" archetypes, we can apply keywords automatically ("auto-keywords") by analyzing the *purpose*, *concept description* and *use*. We add the number of identical "auto-keywords" with a specific weight w_2 to the scoring. The keywords are weighted too, by the w_1 constant.

Note that only one of both should be used. In many cases, an archetype's keywords are the same as the "auto-keywords", which can double the similarity score. So only two cases are foreseen: ($w_1=1$ and $w_2=0$) or ($w_1=0$ and $w_2=1$).

$$score = N_{kw} * w_1 + N_{akw} * w_2$$

Where:

- *score*: Total score for the similarity of the archetypes.
- N_{kw} : Number of identical keywords.
- w_1 : Weight for the keywords.
- N_{akw} : Number of identical "auto-keywords".
- w_2 : Weight for the auto-keywords.

Level 3 – Item Labels

Similar to the keyword approach, it may also be useful to use the labels of data elements. Data elements have names that are mostly specific to their use and can create a link between related archetypes. If the number of same labels is added to the equation, the weighted sum of keywords and labels results in the similarity score:

$$score = N_{kw} * w_1 + N_{akw} * w_2 + N_{il} * w_3$$

Where:

- *score*: Total score for the similarity of the archetypes.
- N_{kw} : Number of identical keywords.
- w_1 : Weight for the keywords.
- N_{akw} : Number of identical "auto-keywords".
- w_2 : Weight for the auto-keywords.
- N_{il} : Number of identical item labels.
- w_3 : Weight for the item labels.

For this approach, it is important to exclude labels that are too common. For this project, the following labels have been identified as common and unspecific:

- tree
- comment
- extension
- item tree
- any event
- description
- event series
- additional details

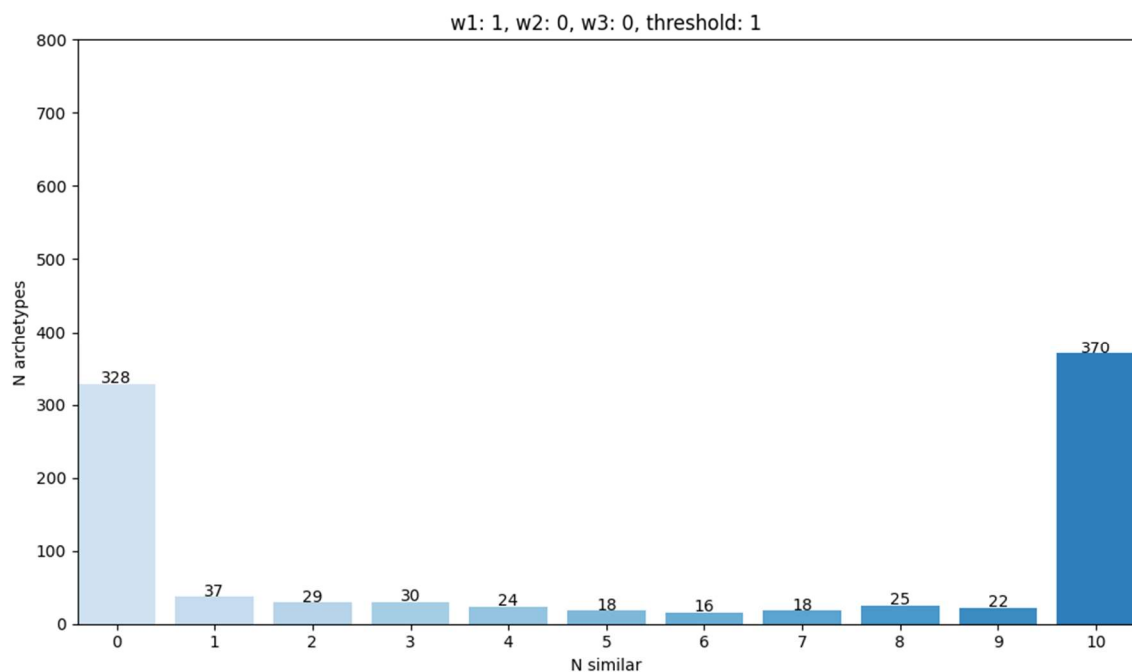
When performing this scoring, these item labels should be ignored.

Results of the Recommendation Approach

The scoring of similarity between archetypes was realized as described above, including three parameters to create a content-related link. After finding the similar archetypes, all children, parent, excluded, and included archetypes were removed, as these are already identified as related in the corresponding attribute of the data set objects.

To avoid overwhelming users with too many suggestions, only the "Top 10" similar archetypes were added to the "similar" list item of the archetype in the data set.

To set a baseline for the links between archetypes, a list was made of the number of similar archetypes found for one archetype and the number of archetypes that have exactly this number of similar archetypes. The following graph shows the distribution for the case where archetypes are only linked by their keywords:



Number of similar archetypes vs number of archetypes, linked by keywords.

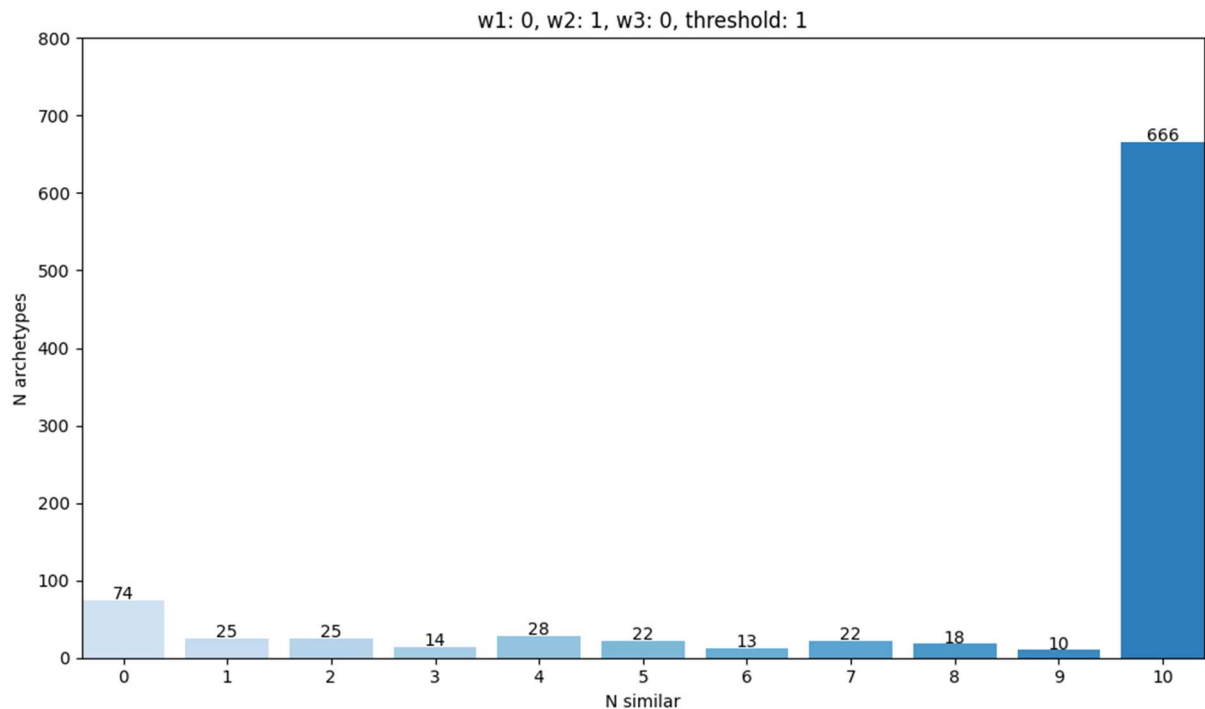
As shown in the graph, there are **328** archetypes in the "**0 similar**" group, while there are **370** archetypes in the "**10 similar**" group.

The similarity scoring aims to achieve two things:

1. **Reduce** the number of archetypes in the "**0 similar**" group.
2. **Avoid** opening the definition of similarity so much that saturation occurs (i.e., all archetypes fall into the "**10 similar**" group).

It should be remembered that semantic linking will never be completely homogeneous. There will always be some archetypes in the "0 similar" group and some in the "10 similar" group.

When using "auto-keywords" for linking archetypes, in addition to the **~290** archetypes without keywords, a shift in the similar archetypes can be seen:



Number of similar archetypes vs number of archetypes, linked by auto-keywords.

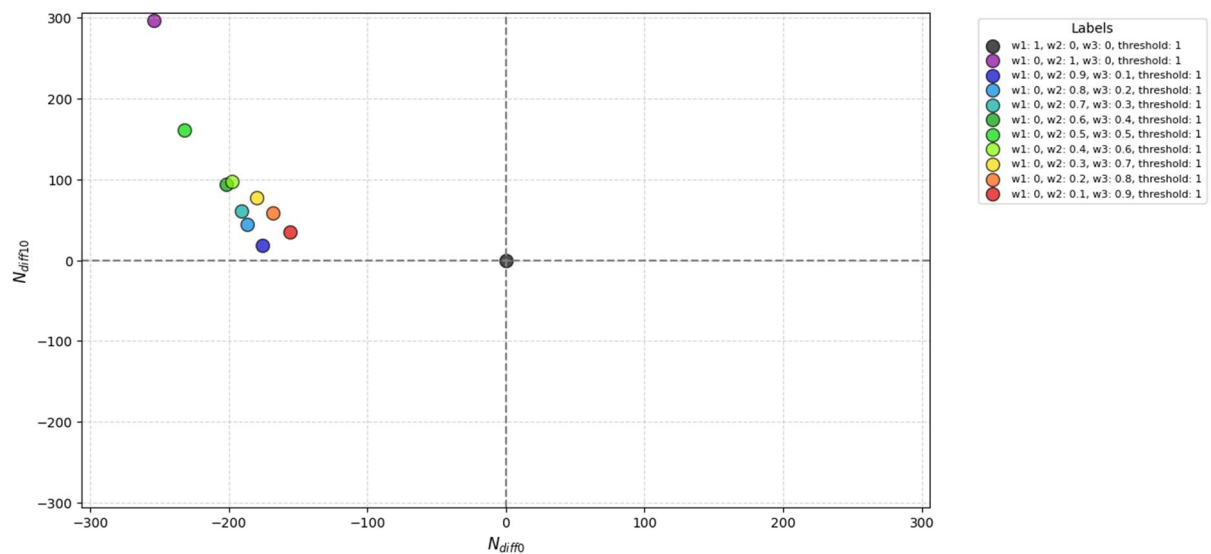
Now only **74** archetypes remain in the **"0 similar"** group. On the other hand, the number of archetypes in the **"10 similar"** group has **increased dramatically**. The next step is to see if this change can be controlled by adding other similarities, such as the item labels used.

With the goal of reducing the number of archetypes in the "0 similar" group and reducing or maintaining the number in the "10 similar" group, a comparison can be made to understand the influence of the weight parameters.

For the comparison, the number of archetypes in the "0 similar" group was subtracted from the 328 of the default case (N_{diff0}). It would be beneficial to have a negative N_{diff0} , as this means there are fewer "0 similar" archetypes.

Similarly, the number of archetypes in the "10 similar" group was subtracted from the 370 of the default case (N_{diff10}). Here, it would be beneficial to maintain the default value ($y=0$) or have a negative N_{diff10} . A negative value would show the redistribution of similarities to other archetypes.

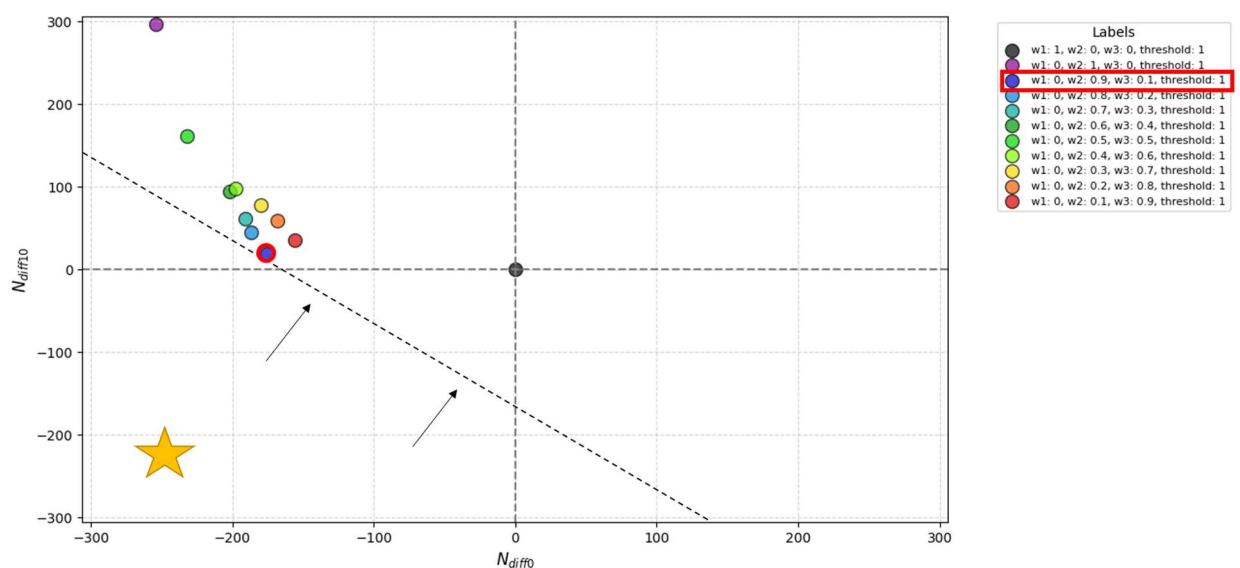
In the following graph, the combination between "auto-keywords" and item labels was explored:



Comparison of combinations between auto-keywords and item labels.

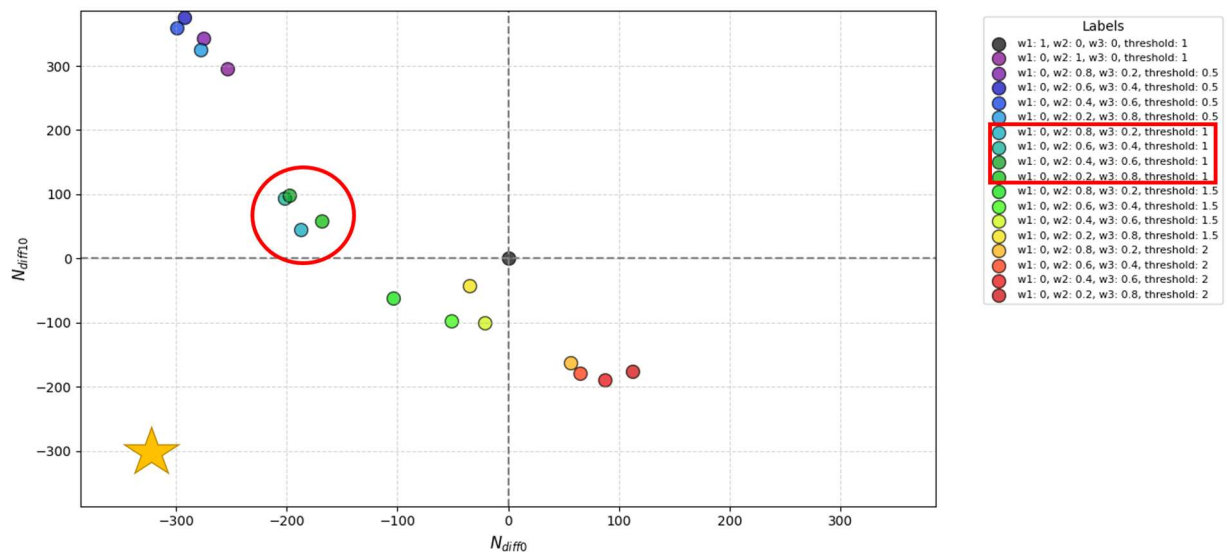
In this graph, the "only keywords" approach is in the middle (0,0). As can be seen, all combinations diminish the number of archetypes in the "0 similar" group but tend to saturate the "10 similar" group.

Starting with the proposition that the ideal combination of these variables would provide a lower "0 similar" and "10 similar" archetype count, the ideal combination would be situated in the lower left quadrant of the graph (depicted as a star in the following graph). If equal importance is given to lowering both categories, a limit can be drafted with a slope of -1. By moving this limit upwards, the "ideal" combination is the first touched by the limit. In our case, this is the combination of $w_2=0.9$ and $w_3=0.1$.



Selection of the best combinations between auto-keywords and item labels.

In the previous comparison, the threshold was set to "1", to require a minimum of identical keywords and item labels. To ascertain the influence of the threshold, the following combinations were graphed:

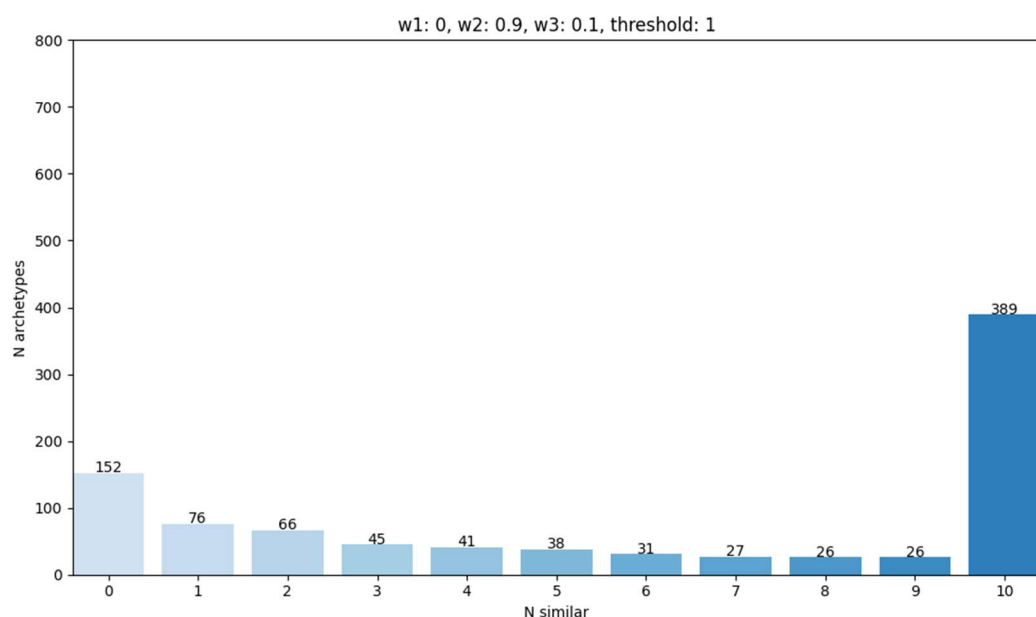


*Influence of threshold on combinations between auto-keywords and item labels.
Most threshold=1 cases are situated as indicated.*

From this graph, it can be seen that a high threshold tends to create more archetypes in both the "0 similar" and "10 similar" groups, while a low threshold saturates the "10 similar" group and diminishes the "0 similar" group. A threshold of "1" seems appropriate for this combination of keywords and item labels.

Conclusion: We will apply the formula $\text{score} = 0.9 \times N_{akw} + 0.1 \times N_{il}$ for scoring. Only scores with ≥ 1 will be considered as recommendations.

The distribution of similar archetype amount for this equation is depicted in the next graph:



Number of similar archetypes vs number of archetypes, linked using the resulting equation.

The size of the "0 similar" group has been reduced from the baseline **328 to 152**.

Note that the similar archetypes found are saved in the data set with every archetype under the *similar* key.

Considerations for the Future

As we look ahead, several enhancements could further improve the recommendation algorithm and overall system effectiveness:

- Incorporate more pattern knowledge from modellers into the recommendation algorithm to enable a dynamic suggestion system.
- Add the archetype class as an additional weighted variable in the similarity score calculation.
- Identify and group "categories" based on keywords to enhance semantic organization.
- Explore the use of AI techniques to automatically group archetypes.

References

- Swartz, Luke. (2003). Why People Hate the Paperclip: Labels, Appearance, Behavior, and Social Responses to User Interface Agents. 10.13140/RG.2.1.2508.1047.

8.8. Project Publication and Source Code

This fellowship project is publicly available on GitHub:

https://github.com/martinkochdesign/archetype_companion

Source Code

The complete source code for the ADL extraction component can be found in the *src* folder of the repository. Please note that the version number is included in the folder or file names (for example, **V4_11** stands for version 4.11).

Executable Release

For users who prefer not to run the source code directly, a compiled executable version of the script is provided in the *dist* folder. This executable is designed to work on any modern Windows computer, making it easy to get started without additional setup.

Because the executable file is bigger than 100MB, the file cannot be pushed directly to the GitHub repository. For larger files, **Git Large File Storage** (or Git LFS) has to be employed.

For this, the current version at the time (v3.7.0) from Git LFS (<https://git-lfs.com/>) was installed on the computer with the GitHub Desktop installation.

In the `.gitattributes` file of the repository, a line was added to manage the large `.exe` files

```
# Auto detect text files and perform LF normalization
* text=auto
# Manage large exe files with Git LFS
*.exe filter=lfs diff=lfs merge=lfs -text
```

8.9. Automation

As this project is hosted on GitHub, we can take advantage of **GitHub Actions** to run the ADL extraction script automatically and periodically. For this purpose, *headless mode* was enabled, and the download direction was set to the international CKM as explained in [GUI](#).

This version of the script, with headless mode enabled, is saved as **adl_extraction_tool.py** in the root of the project. The change of name is done to avoid confusing the source code with the live script.

Requirements

The **import** section of the Python code lists all libraries required by the script:

```
import datetime
import requests
import os
import zipfile
import re
import json
import pandas as pd
import tkinter
import customtkinter
from customtkinter import FileDialog
import shutil
import time
```

For execution of the script in GitHub Actions, a list of libraries is needed for their installation. Not all libraries from the import section are listed here, because Python's standard library already includes several of them. The **requirements.txt** file, located in the root of the repository, contains the following:

```
requests
datetime
pandas
customtkinter
```

Automation YAML File

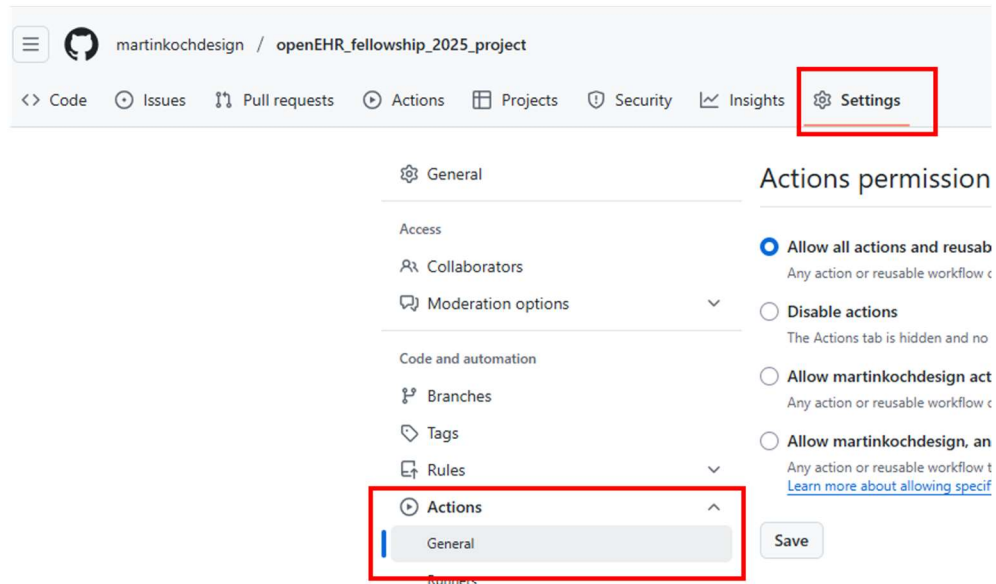
For automation, GitHub Actions requires a YAML file with instructions. In this project, the YAML file is called **update-dataset.yml** and is located in `.github/workflows` in the repository. This is the content of the YAML file used:

```
name: Weekly Archetype Dataset Update
on:
  schedule:
    - cron: '0 3 * * 1' # Runs every Monday at 03:00 UTC (see https://crontab.guru/ for
configuration)
  workflow_dispatch:    # Allow manual trigger from GitHub UI
jobs:
  update-dataset:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repository
        uses: actions/checkout@v3
      - name: Set up Python
        uses: actions/setup-python@v4
        with:
          python-version: '3.11'
      - name: Install Python dependencies
        run: |
          if [ -f requirements.txt ]; then pip install -r requirements.txt; fi
      - name: Run extractor script
        run: |
          python automation_test.py
      - name: Commit and push updated dataset
        run: |
          git config --global user.name "github-actions"
          git config --global user.email "github-actions@github.com"
          git add dataset.js
          git commit -m "Automated weekly dataset update" || echo "No changes to
commit"
          git push
```

This automation is executed periodically according to the schedule. The schedule can be changed as needed; see *Crontab Guru* (<https://crontab.guru/>) for more information on *cron* syntax.

Additional Setup in GitHub

Because the automated script writes files to the GitHub repository, the repository settings must be adjusted to allow this. This setting can be found in the repository's **Settings** tab under **Actions > General**.



Action settings in GitHub.

Here, the **Read and write permissions** option should be selected.

Workflow permissions

Choose the default permissions granted to the GITHUB_TOKEN when running workflows in this repository. You can specify more granular permissions in the workflow using YAML. [Learn more about managing permissions.](#)

☒ **Read and write permissions**
Workflows have read and write permissions in the repository for all scopes.

☐ Read repository contents and packages permissions
Workflows have read permissions in the repository for the contents and packages scopes only.

Choose whether GitHub Actions can create pull requests or submit approving pull request reviews.

☐ Allow GitHub Actions to create and approve pull requests

Read and write permissions.

Do not forget to save after changing the permissions.

8.10. Detailed description of use

For a detailed description of use for the archetype ADL1.4 data extraction tool, please refer to [ANNEX I - ADL Extraction Tool – User Manual](#).

9. Visualization Tool – The Archetype Companion

9.1. Introduction

Once the archetype information has been extracted in the *dataset.js* file, the second part of the project begins.

The general idea is to design a tool that enhances the way to search, plan, and visualize openEHR archetypes, providing a comprehensive platform for clinical knowledge modellers to efficiently manage archetype data.

General Description

The Archetype Visualization and Planning Tool aims to provide detailed information on existing archetypes and facilitate the addition of future archetypes to the ecosystem.

The main **development goals** are:

- Provide comprehensive information on existing archetypes from an archetype source.
- Facilitate the fast search of existing archetypes.
- Help to understand the archetype ecosystem by network visualization.
- Enable the planning of addition of new archetypes to the ecosystem.

The **key features** can be broken down into:

- Archetype Search
- Archetype Project Management
- Network Visualization

The following sections provide detailed descriptions of the key features as currently envisioned. The proposal is made with the best intentions but should be taken "as is." Some points may change due to new opportunities or technical limitations found during development.

Key Features

Archetype Search

The archetype search should be a simple search algorithm that takes the data from the dataset and filters out all of the archetypes that do not contain the search text.

The following features for the search should be considered:

- Search input over a text field.
- Search result update at every keystroke in the text field (Does not need a separate "Search" button).
- Search should be fast.
- Search archetypes. The search should be done by filtering one or more of the following:
 - keywords
 - class
 - purpose
 - concept name
 - archetype ID
- Search is only executed over the existing archetypes from the "dataset.js" file.

Additional functionalities that might be considered are:

- Adding regex or regex-like expressions to the search algorithm, so wildcards ("*") can be used.
- Sort search results alphabetically (A-Z, Z-A).
- Show number of found results.
- Show preview of the text, where the search text was found.

Archetype Project Management

The project management component of the tool should enable users to manage collections of archetypes and track development progress.

The following features for the project management should be considered:

- Archetype Management
 - Existing Archetypes: Archetypes from the dataset.js file.
 - Display information on archetype after clicking on it. The information of an archetype should include:
 - Archetype ID
 - Concept name
 - Purpose
 - Items
 - Inclusions
 - Exclusions
 - Parent
 - Children
 - Description
 - State
 - Date
 - Keywords
 - Recommended archetypes
 - Alerts for archetype predrafts or drafts usage.
 - Suggestion of similar and relevant archetypes.
 - New Archetypes: Planned archetypes added by the user.
 - Only outlines of archetypes, not fully functional ADL1.4 archetypes.
 - Are added to the collection. They only exist in the collection context.
 - Display editable information on archetype after clicking on it:
 - Archetype name
 - Class
 - Purpose
 - Keywords
 - Assigned elements
 - Included archetypes. Only archetypes from the collection can be included.
 - Collection of Archetypes

- New and existing archetypes should be bundled into a collection by the user.
 - Existing archetypes can be added or removed from the collection.
 - New archetypes are created in the collection and get deleted when removed from the collection.
- Checklist Management
 - Structured Checklist
 - Structured in sections and elements. One level deep (no section within a section).
 - Sections and elements are movable.
 - Sections and elements can be added/deleted.
 - Assigned archetype per element.
 - "Completed" marker per element.
 - "Change request" marker per element.
 - Load/save capabilities in format XLSX.
- Project Management
 - Save/load capabilities in JSON format.
 - Saves project metadata, checklist data, new archetypes, and archetype collection.
 - Delete project data.

Additional functionalities that might be considered are:

- Project metadata might include:
 - Author name
 - Author affiliation
 - Copyright
 - Description
 - Date of generation
- New Archetype
 - Auto-suggest from keyword pool when typing in keywords.

Network Visualization

The following features for the network visualization should be considered:

- A network visualization using nodes (circles) and edges (connecting lines).
- Clicking on a node should select it and display its information for convenience.
- Basic navigation around the network should be implemented: Move and zoom.
- Visualization Modes:
 - Network visualization of archetypes and their connections.
 - Visualize parent-child relationship (specialized archetypes).
 - Visualize inclusion/exclusion relationships.
 - Network visualization of archetype collection.
 - Visualize new and existing archetypes.
 - Visualize inclusion if exists between members of collection.
 - Visualize specialization if exists between members of collection.
 - Visualization of recommended or similar archetypes.
 - Visualization of project/checklist.
 - Show sections.
 - Show elements of sections and connect to corresponding section.
 - Show archetypes from collection, if they have been assigned to an element. Show the connection between archetype and element.

Other Requirements

Apart from the key functionalities, a few additional requirements can be already determined.

- About Page
 - The About page will provide an overview of the tool, its purpose, and the team behind its development.
- Accessibility:
 - Cross-platform use (via web browser): Ensuring cross-platform compatibility will allow users to access the tool from various devices and browsers, enhancing usability and reach.
 - No user account management: By not requiring user account management, the tool simplifies access and reduces potential security concerns related to user data storage.
- User Interface and Experience:
 - Intuitive navigation and responsive design to enhance user experience.

- Incorporate user feedback mechanisms for continuous improvement.
- Performance:
 - Optimize load times and responsiveness, especially when handling large datasets.
- Integration:
 - Support integration with other systems or tools as needed.
- Documentation and Support:
 - Provide comprehensive documentation and support resources to assist users.

These requirements might change over time, but special care is going to be given to fulfil these development goals.

9.2. User Journey Mapping

To enhance the understanding of usability requirements, a user journey mapping exercise was conducted. This involved documenting a first-person user journey to illustrate how the application is intended to be used and to identify potential usability bottlenecks.

User Persona: Clinical Data Modeller

The target user for this application is a *clinical data modeller* experienced in the *openEHR* framework. While not requiring extensive programming knowledge, the user should be familiar with data structuring and template planning principles.

The user has compiled a list of data elements, likely already organized, and needs to plan the structuring of templates using appropriate archetypes. Their goal is to adhere to the openEHR philosophy of reuse, sharing, and collaboration on archetypes to achieve advanced interoperability on an international scale.

This user journey mapping aims to understand the user's journey, identify any challenges they might face, and ensure the tool meets their needs effectively.

User Journey

The following steps outline a typical user journey from the beginning to the end of the planning process:

1. Open the Application

The user opens the application for the first time. The **main view** opens. The user does not know how to use the application, so they open a link to a user manual.

- Comments for UX design:
 - The application should be easy to open.
 - If opened for the first time, prompt the user to create a new project (or create a new project automatically).
 - Show the project name in the **main view**.
 - Show a **splash screen** if this is a new project, with a link to the user manual.
 - Add a link to a manual on the **About page** of the application.
 - Evaluate if there is a need for a tutorial in the application.
 - Main view should show:
 - Overview of collection
 - Checklist overview
 - search view
 - Visualization
 - Archetype info

2. About the Application

The user wants to know more about the application. They open the **about page** of the **main view** for information about data security, authorship, and more.

- Comments for UX design:
 - Create an independent **about page**. For easy access, add a button to the **main view**.
 - Include in the **about page**:
 - Legal information (copyright, license, authorship).
 - Link to more information about openEHR, this project, and Catsalut.
 - Link to a user manual.
 - Declaration about data usage and privacy.

3. Edit/Rename Project

The user opens the settings of the project and changes the name and adds metadata in the **project options view**.

- Comments for UX design:
 - Allow easy editing of project metadata.
 - Provide a dedicated window for editing project metadata.

- For easy access, add a button to the **main view**.

4. Add Data Elements to/Edit the Checklist

The user adds elements to the checklist in the **checklist editor view**. The user can save and load this checklist from an Excel file.

- Comments for UX design:
 - Provide clear instructions on the format for XLSX checklists.
 - Allow manual addition and editing of checklist items at both element and section levels.
 - Enable loading and saving of the checklist in XLSX format.
 - Consider a dedicated window for checklist editing.
 - Provide direct access to the editor via the **checklist sidebar**.

5. Search for Archetypes

The user can scroll through a list of "existing" archetypes in the **archetype list sidebar**. If the user wants to filter or search this list for applicable archetypes, they access a **search view**. The search is conducted with a search term that should be searched in different attributes of all "existing" archetypes. While searching, the user references the element checklist. If an archetype is selected, the relationship of this archetype and other archetypes should be shown in the **visualization view**. This gives the user the possibility to explore related archetypes. The user, having gone to an archetype that is not of interest, can open a search **history** of visited archetypes to navigate back to previously seen archetypes.

- Comments for UX design:
 - Let the user select attributes to be searched (class, ID, purpose, items, keywords).
 - Add button to open history window.
 - Display the **checklist sidebar** for easy reference.
 - Display the archetype relationships in the **visualization view**.
 - Change selected archetype by clicking on a node in the **visualization view**.
 - Show the information of the selected archetype in the **info view**.
 - Show the number of found archetypes in the search results.
 - Have a "back button" to revisit earlier finds. Maybe have a detailed list of the last archetypes visited.
 - The **search view** should include:
 - Text field for search term.
 - Selectors for attributes to be searched.
 - List of filtered "existing archetypes".

6. Navigate Similar Archetypes

The user wants to navigate from the selected archetype to recommended or similar archetypes. For this, the user changes the **visualization view** to show the related archetypes. The network can be navigated by clicking on the archetype nodes.

- Comments for UX design:

- Have an option/selector in the **visualization view** to show the recommended archetypes as a network.

7. Add Archetype to Collection

When the user has found an appropriate archetype, they add the selected archetype to their **collection view**.

- Comments for UX design:
 - Show a list of collection items. Make the list selectable, so archetypes can be selected and visualized in the **info view** and **visualization view**.
 - The archetype is selected from the "existing archetype" list, therefore the element for adding it should be near or in the **search view**.

8. Create/Edit New Archetype

The user detects that there are data elements in their checklist that cannot be covered by existing archetypes. To create a placeholder "new archetype", the user adds a "new archetype" to the **collection view** via a **new archetype editor view**.

- Comments for UX design:
 - Create the new archetype in a specific **new archetype editor view**.
 - As the "new archetypes" only exist in the collection, the creation should be initiated from the **collection view**.
 - While creating/editing the new archetype, the **checklist sidebar** should be visible.
 - Once created, show the info of the new archetype in the **info view**.
 - The **new archetype editor view** should be opened to edit the archetype from the **info view**.
 - The attributes for a "new archetype" should include:
 - ID.
 - Keywords.
 - Purpose.
 - Inclusion statements for other archetypes of the collection.
 - Data elements from the checklist assigned to this archetype.

9. Remove "New" or "Existing" Archetype from Collection

The user has added an archetype to the collection but realizes that the archetype does not belong in the **collection view**. When the archetype is selected, the user presses a button to delete the archetype from the collection.

- Comments for UX design:
 - A user input element (i.e., button) should be added to the **collection view** to delete archetypes from the collection.

10. Visualize Collection

The user wants to see the relationship between archetypes in the collection.

- Comments for UX design:
 - Have an option/selector in the **visualization view** to show the collection as a network.

- While seeing the network, the **collection view** and **info view** should be visible.

11. Mark Data Element as Resolved/Assign Archetype to Checklist Element

To edit the checklist items, the user enters the **checklist editor view**. Here the user marks a data element as resolved (so it does not have to be searched for anymore) and can assign an archetype from the collection. If the user thinks that the assigned archetype needs a change request, this can be marked here as well. The user needs to see the **info view** for reference.

- Comments for UX design:
 - Have the archetype **info view** visible.
 - For every element, there should be the following information:
 - Resolved (Yes/No)
 - Data element name
 - Assigned archetype
 - Needs change request (Yes/No)
 - Data elements and sections should be movable.

12. Save Project

The user saves the project to not lose progress. Saving should be done frequently, so the user needs instant access.

- Comments for UX design:
 - The save button should be located in the **main view** for fast access.
 - Additionally, a save button with the same functionality could be provided in the **project options view**.
 - The project should be saved to a JSON file and should contain:
 - Project metadata
 - Collection
 - "New archetypes"

13. Load Project

The user wants to resume a previous session by loading a project JSON file. This action will potentially delete current project data.

- Comments for UX design:
 - Ask the user if they are sure before deleting existing data.
 - This is not a frequently used function, so it can be added in the **project options view**.

14. Delete Project / Create a New Project

The user wants to start a new project, deleting all data from the current project.

- Comments for UX design:
 - Ask the user if they are sure before deleting existing data.
 - This is not a frequently used function, so it can be added in the **project options view**.

15. Visualize Project

The user wants to see the connection between data elements and the assigned archetypes.

- Comments for UX design:
 - Have an option/selector in the **visualization view** to show the project as a network.
 - While seeing the network, the **collection view** and **info view** should be visible.

16. Export Project Report

The user wants to create a report that serves as documentation for the ongoing development. The report should include all information, and apart from the checklist Excel, should provide comprehensive project details.

- Comments for UX design:
 - This is not a frequently used function, so it can be added in the **project options view**.
 - The format of the report should be compatible (HTML, PDF, etc.).

Through this structured user journey mapping, we aim to enhance the tool's usability and ensure it aligns with user expectations and needs.

Summary of Design Requirements for Different Views/Pages/Windows

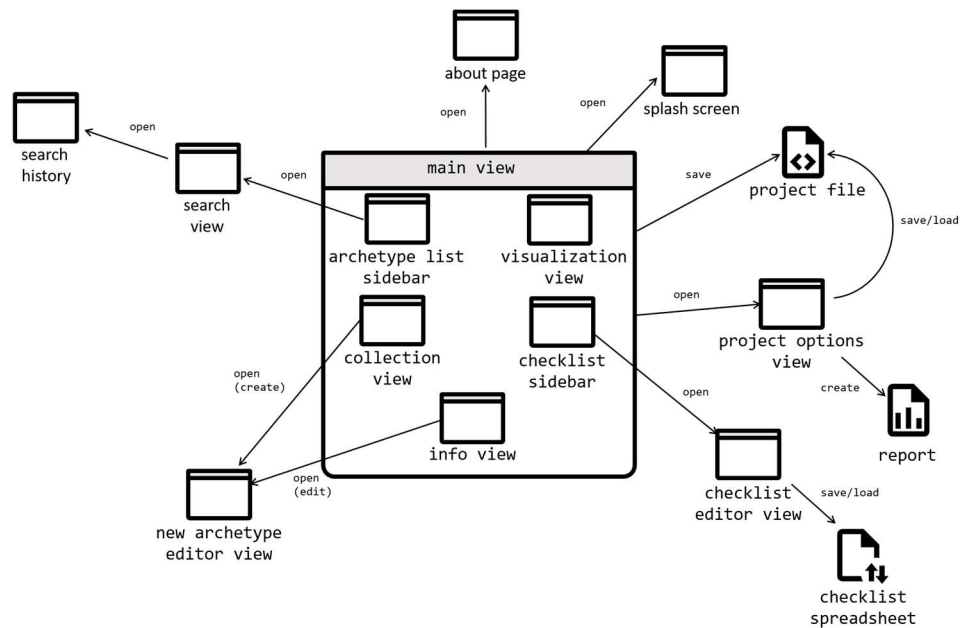
Using this user story, a list of necessary pages/windows/views can be summarized:

- **Main View**
 - Opens on start
 - Display project name
 - Direct access to
 - About Page
 - Project Options View
 - Save Project
 - Includes at start
 - Checklist Sidebar
 - search view
 - Visualization View
 - Info View
 - Collection View
- **Splash Screen**
 - Opened automatically from Main View
- **About Page**
 - Called from Main View
 - Contains

- Legal information (copyright, license, authorship).
 - Link to more information about openEHR, this project, and Catsalut.
 - Link to a user manual.
 - Declaration about data usage and privacy.
- **Project Options View**
 - Called from Main View
 - Edit project metadata
 - Function to
 - Save Project
 - Load Project
 - New Project
 - Create Project Report
- **Checklist Editor View**
 - Called from Checklist Sidebar
 - Provides for every element the following information:
 - Resolved (Yes/No)
 - Data element name
 - Assigned archetype
 - Needs change request (Yes/No)
 - User needs to see the Info View for reference
- **Checklist Sidebar**
 - Shown in Main View
 - Provides access to Checklist Editor View
- **Archetype List Sidebar**
 - Shown in Main View
 - Should include:
 - List of filtered "existing archetypes"
 - Button to open the search view
 - Button to clear filter
 - User needs to see the Checklist Sidebar for reference
- **Search View**
 - Called from archetype list sidebar
 - Should include:
 - Text field for search term
 - Selectors for attributes to be searched
 - List of filtered "existing archetypes"
 - Function to open search history window

- Button to clear filter
 - "Add to collection" button
 - User needs to see the Checklist Sidebar for reference
- **Visualization View**
 - Shown in Main View
 - Selector for:
 - Display the archetype relationships
 - Show the recommended archetypes
 - Show the collection
 - Show the project
 - Change selected archetype by clicking on a node in the view
- **History**
 - Opened from search view
 - Shows the search history
 - History items can be clicked to select archetype
- **Info View**
 - Shown in Main View
 - Show the information of the selected archetype
 - Function to open New Archetype Editor View if the selected archetype is "new"
- **Collection View**
 - Shown in Main View
 - Shows collection of "existing" and "new" archetypes
 - Should have a function to delete an archetype from the collection
 - Should have a function to create a "new" archetype
- **New Archetype Editor View**
 - Called from Collection View for new archetype
 - Called from Info View for editing archetype
 - Opened in floating window
 - Attributes for a "new archetype" should include:
 - ID
 - Keywords
 - Purpose
 - Inclusion statements for other archetypes of the collection
 - Data elements from the checklist assigned to this archetype

In summary the access between the different views can be roughly sketched out as in the following figure:

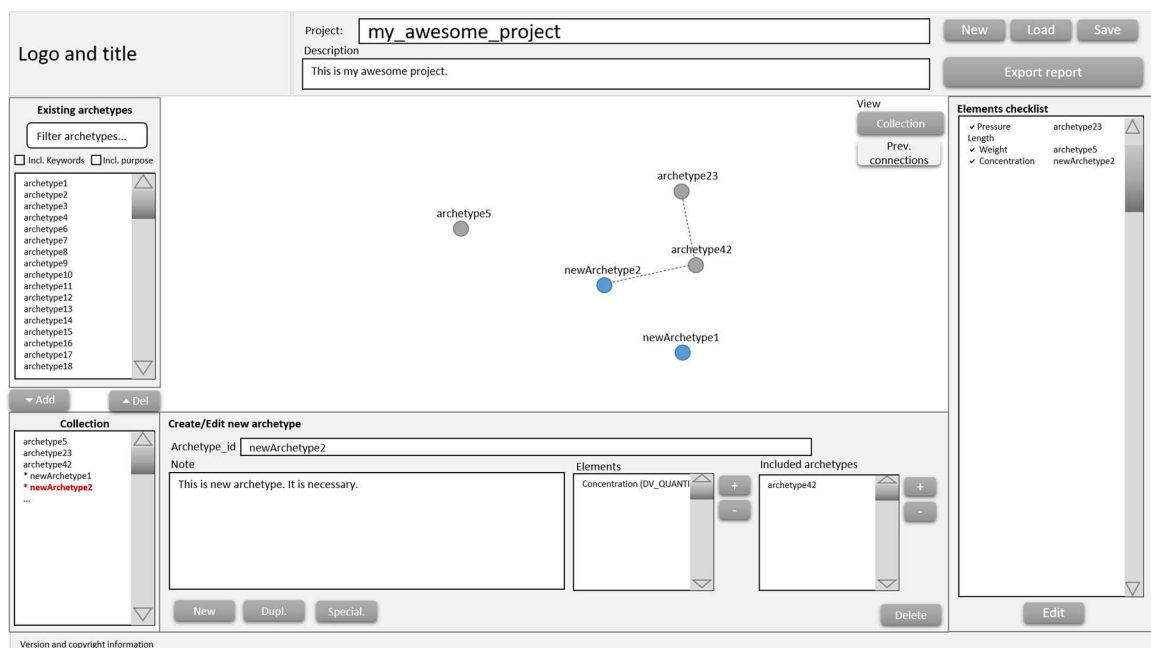


Access between different views/pages/windows and additional files. Note that this schema represents an early stage. The template planner is not represented here,

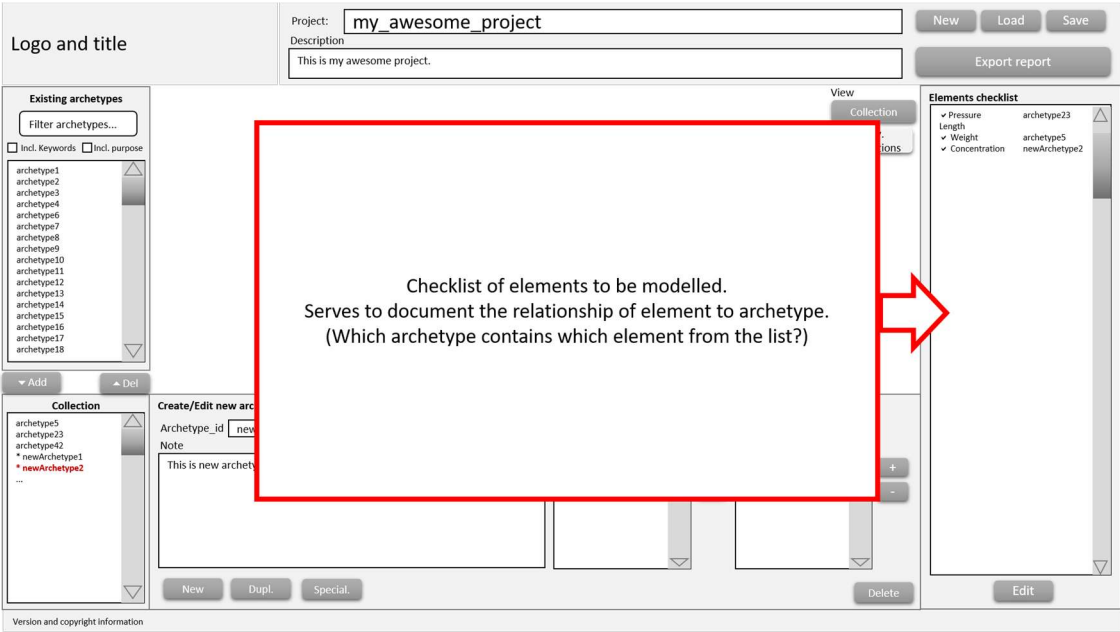
9.3. GUI

Wireframe/Early draft

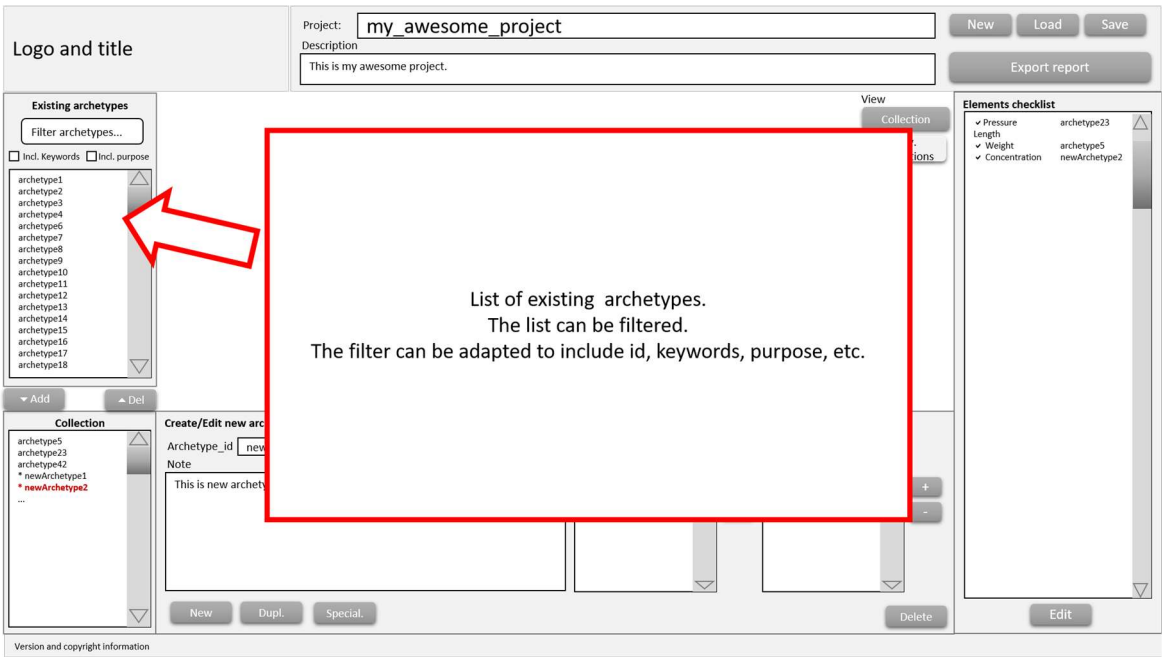
The **Main View** provides access to essential **Project Options** at the top of the interface. On the right-hand side, you will find the **Checklist Sidebar**. On the left, the **Archetype List Sidebar** and **Collection** view are located. A simple filter in the Archetype List Sidebar serves as the **Search** function. The **Visualization** area is situated in the middle section, offering different views to select from.



Main View layout overview



Checklist sidebar



Archetype list

The **Info** panel for a selected archetype is displayed in the lower part of the middle section.

The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

Martin Andreas Koch

The screenshot displays the 'Archetype Companion' interface. At the top, there's a 'Project' field with 'my_awesome_project' and a 'Description' field with 'This is my awesome project.' Below this is a 'Logo and title' section. The main area is divided into several panels. On the left, there's a list of 'Existing archetypes' with a filter button. In the center, a network diagram shows relationships between archetypes (e.g., archetype13, archetype35, archetype8, archetype42, archetype58). On the right, there's an 'Elements checklist' showing a list of elements (Pressure, Length, Weight, Concentration) and their corresponding archetype IDs. A red box highlights the 'Archetype Info' panel, which displays details for 'archetype8', including its concept ('Blood sugar assessment'), purpose ('To assess diabetes mellitus'), and keywords ('Blood sugar, diabetes'). A red arrow points from the network diagram to this panel. Another red arrow points from the 'Elements checklist' to the 'Archetype Info' panel. A third red arrow points from the 'Existing archetypes' list to the 'Archetype Info' panel. A fourth red arrow points from the 'View' button to the 'Archetype Info' panel. A red box contains the text: 'After selecting an archetype, all the archetype information is shown, and the connection between different archetypes is shown in the visualization'.

Project: my_awesome_project
Description: This is my awesome project.

Logo and title

Existing archetypes
Filter archetypes...
Incl. Keywords Incl. purpose
archetype1
archetype2
archetype3
archetype4
archetype6
archetype7
archetype8
archetype9
archetype10
archetype11
archetype12
archetype13
archetype14
archetype15
archetype16
archetype17
archetype18

Collection
archetype5
archetype23
archetype42
* newArchetype1
* newArchetype2

Archetype Info
Archetype_id: archetype8
Concept: Blood sugar assessment
Purpose: To assess diabetes mellitus.
Keywords: Blood sugar, diabetes.

View
Collection
Prev. connections

Elements checklist
✓ Pressure archetype23
✓ Length archetype5
✓ Weight newArchetype2
✓ Concentration

After selecting an archetype, all the archetype information is shown, and the connection between different archetypes is shown in the visualization

Info panel for selected archetype

The screenshot displays the 'Archetype Companion' interface in the 'Archetype collection view'. The top section is identical to the previous screenshot. The main area is divided into several panels. On the left, there's a list of 'Existing archetypes'. In the center, there's a 'Collection' panel with a list of archetypes (archetype5, archetype23, archetype42, * newArchetype1, * newArchetype2). A red box highlights the 'Add new archetype' section, which includes a form for 'Archetype_id' (newArchetype2), 'Note' (This is new archetype. It is necessary.), 'Elements' (Concentration (DV_QUANTI)), and 'Included archetypes' (archetype42). A red arrow points from the 'Existing archetypes' list to the 'Add new archetype' section. A red box contains the text: 'A selected archetype can be added to a collection, meaning it is going to form part of the developed data structure'.

Project: my_awesome_project
Description: This is my awesome project.

Logo and title

Existing archetypes
Filter archetypes...
Incl. Keywords Incl. purpose
archetype1
archetype2
archetype3
archetype4
archetype6
archetype7
archetype8
archetype9
archetype10
archetype11
archetype12
archetype13
archetype14
archetype15
archetype16
archetype17
archetype18

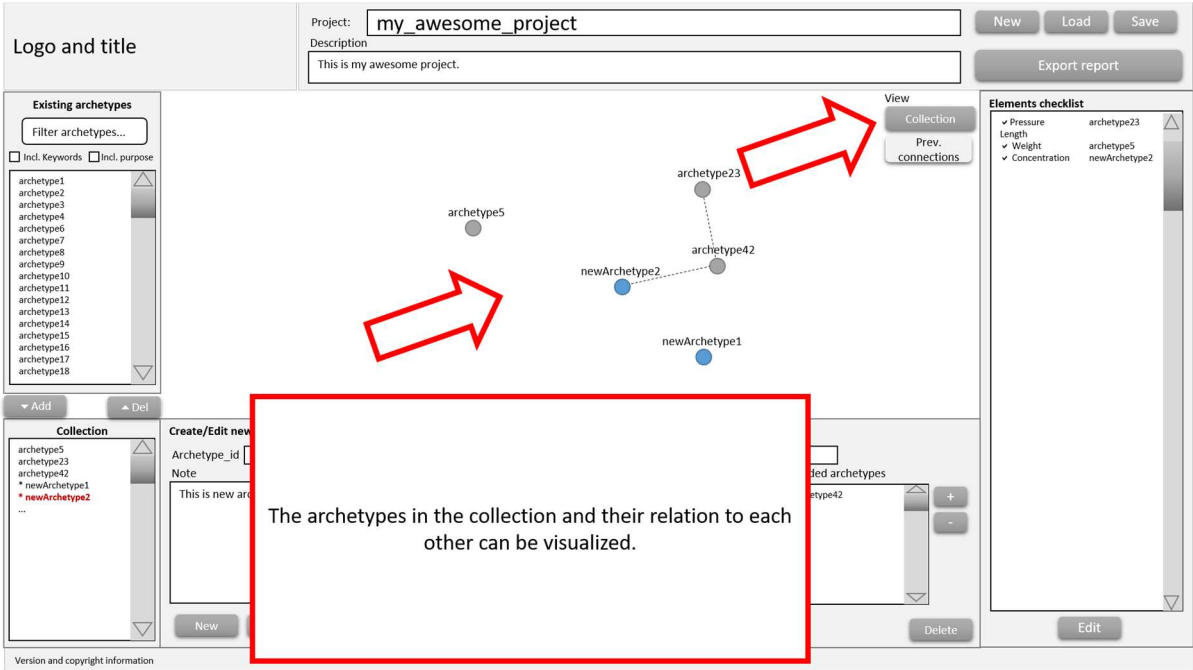
Collection
archetype5
archetype23
archetype42
* newArchetype1
* newArchetype2

Add new archetype
Archetype_id: newArchetype2
Note: This is new archetype. It is necessary.
Elements: Concentration (DV_QUANTI)
Included archetypes: archetype42

New Dupl. Special. Delete Edit

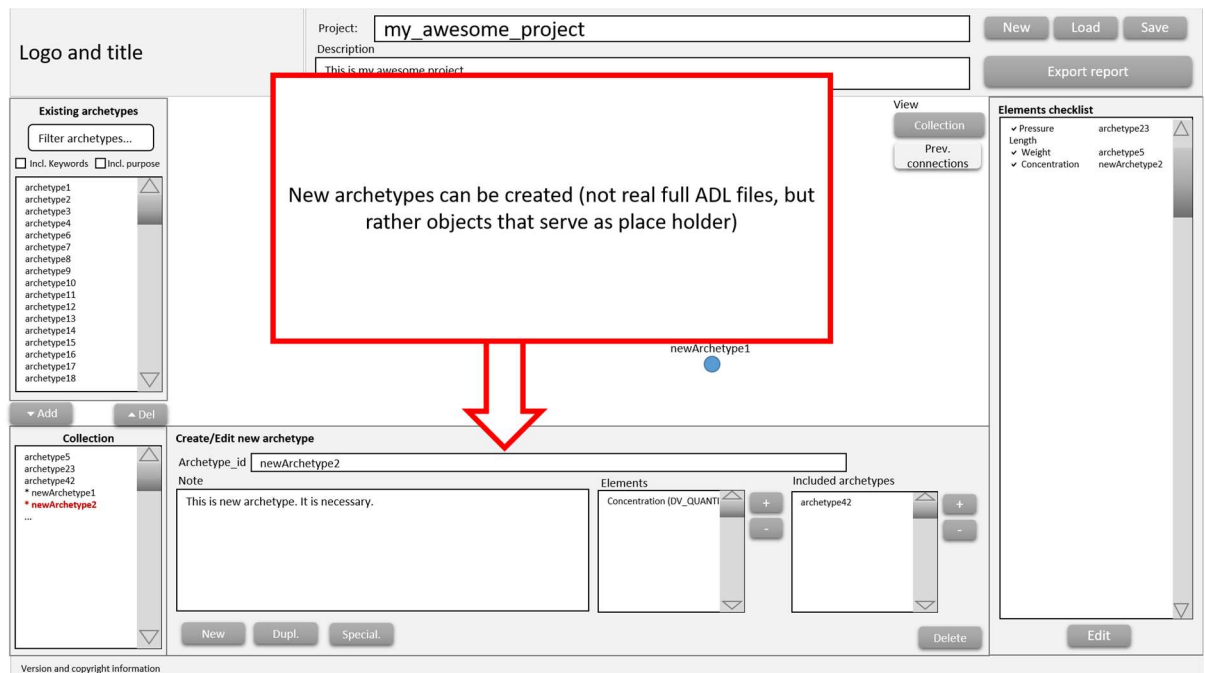
A selected archetype can be added to a collection, meaning it is going to form part of the developed data structure

Archetype collection view

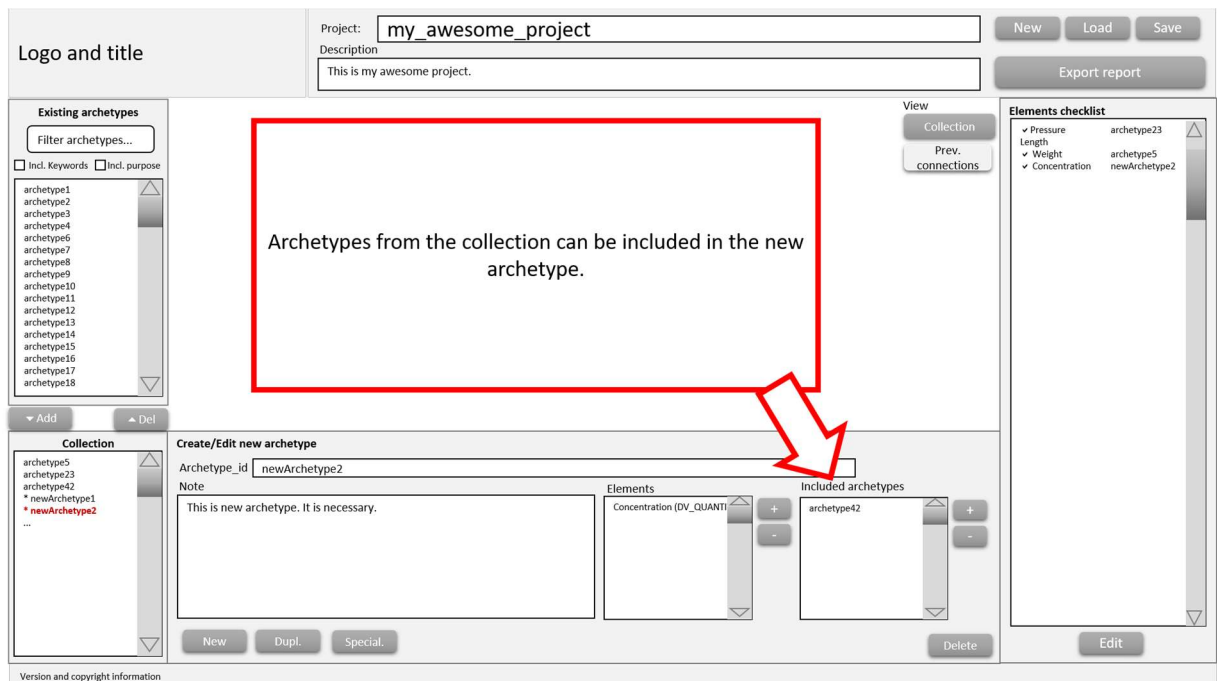


Visualization view collection

For editing or creating new archetypes, the **New Archetype Editor** is shown in the lower part of the middle section.



New Archetype Editor interface



Editor details and options

The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

Martin Andreas Koch

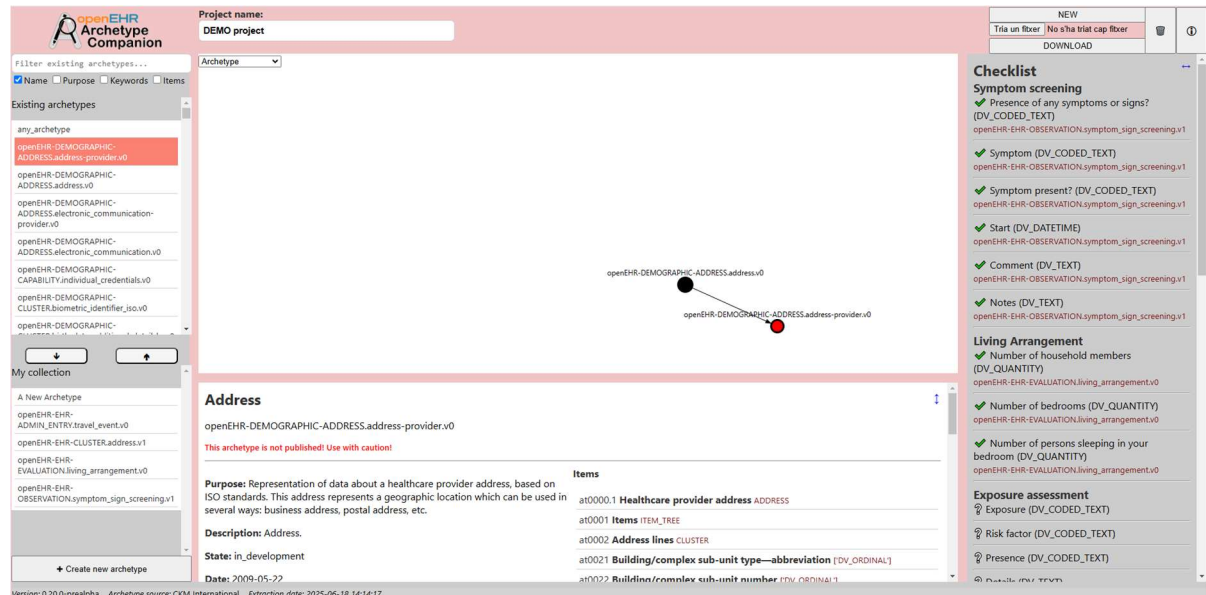
The screenshot shows the Archetype Companion interface. At the top, there's a 'Project' field with the value 'my_awesome_project' and a 'Description' field with the text 'This is my awesome project.' Below these are buttons for 'New', 'Load', 'Save', and 'Export report'. On the left, there's a section for 'Existing archetypes' with a list of archetype IDs (archetype1 through archetype18) and a 'Filter archetypes...' button. Below this is a 'Collection' section with a list of archetype IDs (archetype5, archetype23, archetype42, newArchetype1, newArchetype2) and buttons for 'Add' and 'Del'. In the center, there's a large red box containing the text 'Export a report for documentation'. To the right of this box is a 'Create/Edit new' section with a 'Note' field containing the text 'This is new archetype. It is necessary.' and buttons for 'New', 'Dupl.', and 'Special.'. Further right is an 'Elements' section with a list of elements (Concentration (DV_QUANTI)) and buttons for '+', '-', and 'Delete'. On the far right, there's an 'Included archetypes' section with a list of archetype IDs (archetype42) and buttons for '+', '-', and 'Delete'. At the bottom right, there's an 'Elements checklist' section with a list of elements (Pressure, Length, Weight, Concentration) and buttons for 'Edit' and 'Delete'. A red arrow points to the 'View' button in the top right corner.

Export project report

Note: There is no dedicated **About** page, **History** view, **Checklist Editor**, or separate **Search** view planned for this tool.

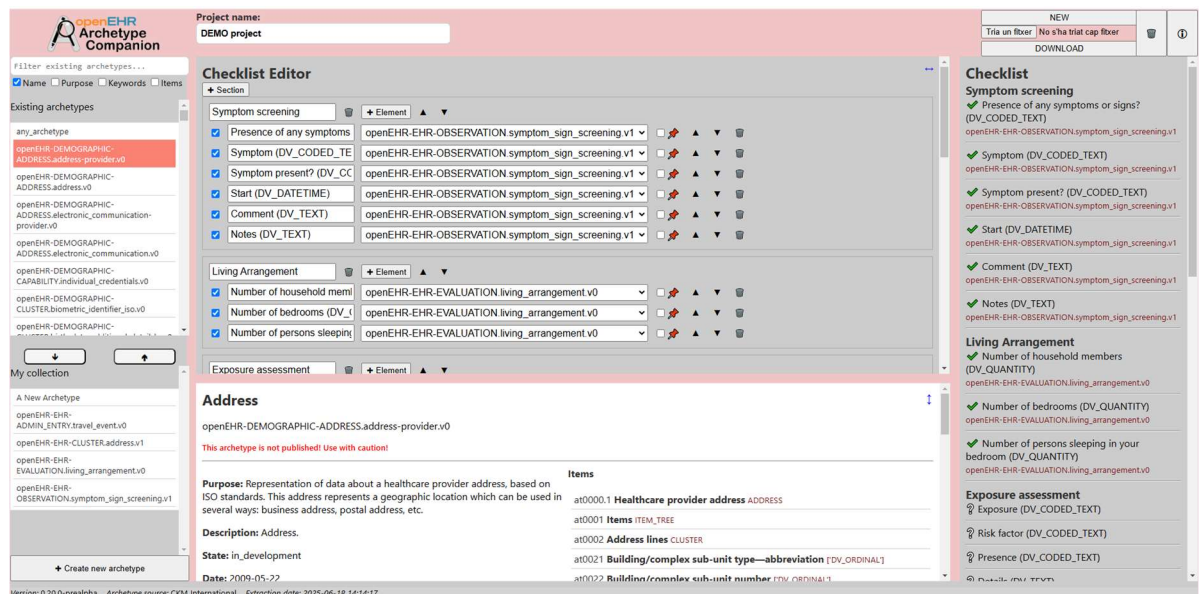
First Iteration of the GUI

Based on the previously defined wireframe, a first iteration of the GUI was implemented. The **Main View** closely resembles the initial wireframe.



First GUI developed - Main View layout.

A separate editing view for the checklist was implemented, as shown in the following figure.



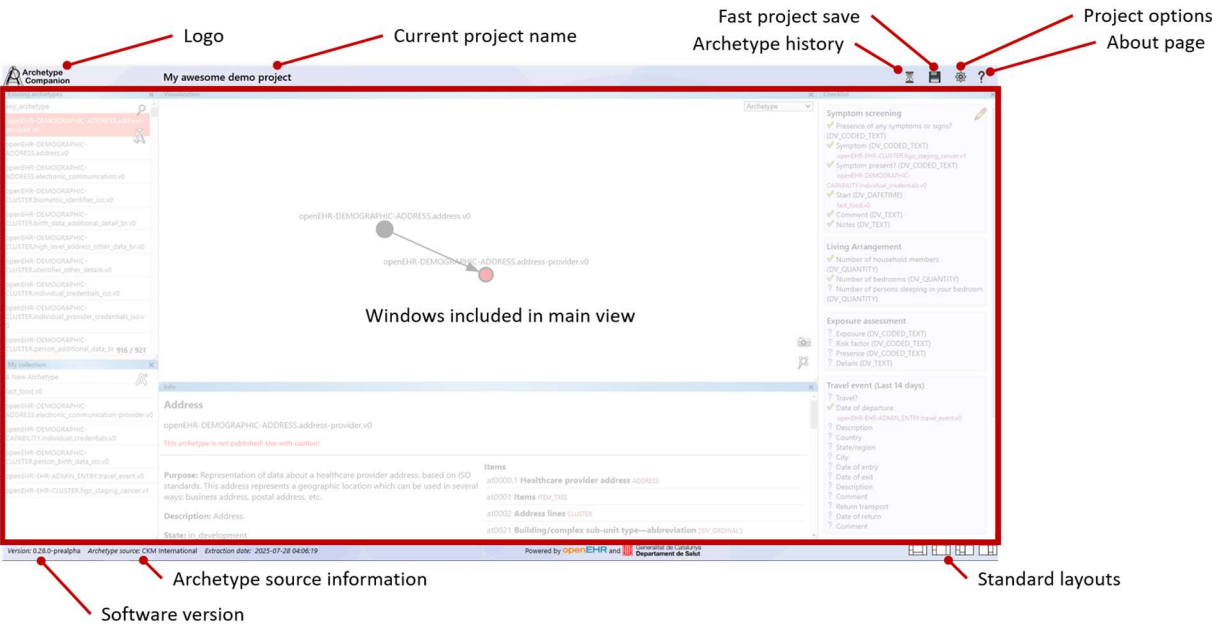
First GUI - Checklist Editor view.

This implementation used a fixed grid layout for the views, without allowing users to resize or reposition them. When integrating results from the User Journey Mapping, this rigid grid approach proved problematic, especially when adding additional views to the project.

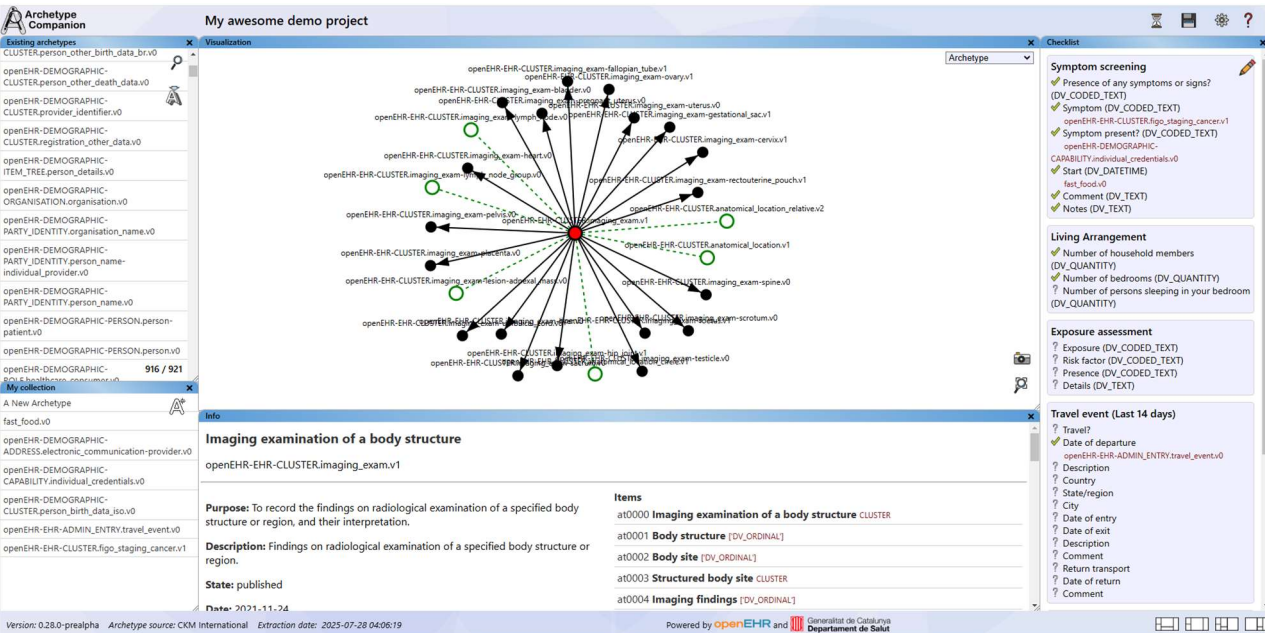
Second Iteration of the GUI

Using the comprehensive list of "Design Requirements for Different Views/Pages/Windows" from the User Journey Mapping, the second iteration of the GUI was drafted.

Main View



Standard view



The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

Martin Andreas Koch

Information view

The screenshot shows the 'Information view' of the Archetype Companion. The main window displays the 'Imaging examination of a body structure' archetype. The left sidebar lists various archetypes, including 'openEHR-DEMOGRAPHIC-CLUSTER.person_other_birth_data.v0' and 'openEHR-DEMOGRAPHIC-CLUSTER.person_other_death_data.v0'. The main content area shows the 'Imaging examination of a body structure' archetype with its purpose, description, state, date, keywords, parent, children, and items. The 'Items' list includes 'at0000 Imaging examination of a body structure CLUSTER', 'at0001 Body structure [DV_ORDINAL]', 'at0002 Body site [DV_ORDINAL]', 'at0003 Structured body site CLUSTER', 'at0004 Imaging findings [DV_ORDINAL]', 'at0005 Additional details CLUSTER', 'at0006 Impression [DV_ORDINAL]', and 'at0007 Comment [DV_ORDINAL]'. The right sidebar shows a 'Checklist' with sections for 'Symptom screening', 'Living Arrangement', 'Exposure assessment', and 'Travel event (Last 14 days)'. The bottom status bar indicates 'Version: 0.28.0-prealpha', 'Archetype source: CKM International', 'Extraction date: 2025-07-28 04:06:19', and 'Powered by openEHR and Generalitat de Catalunya Departament de Salut'.

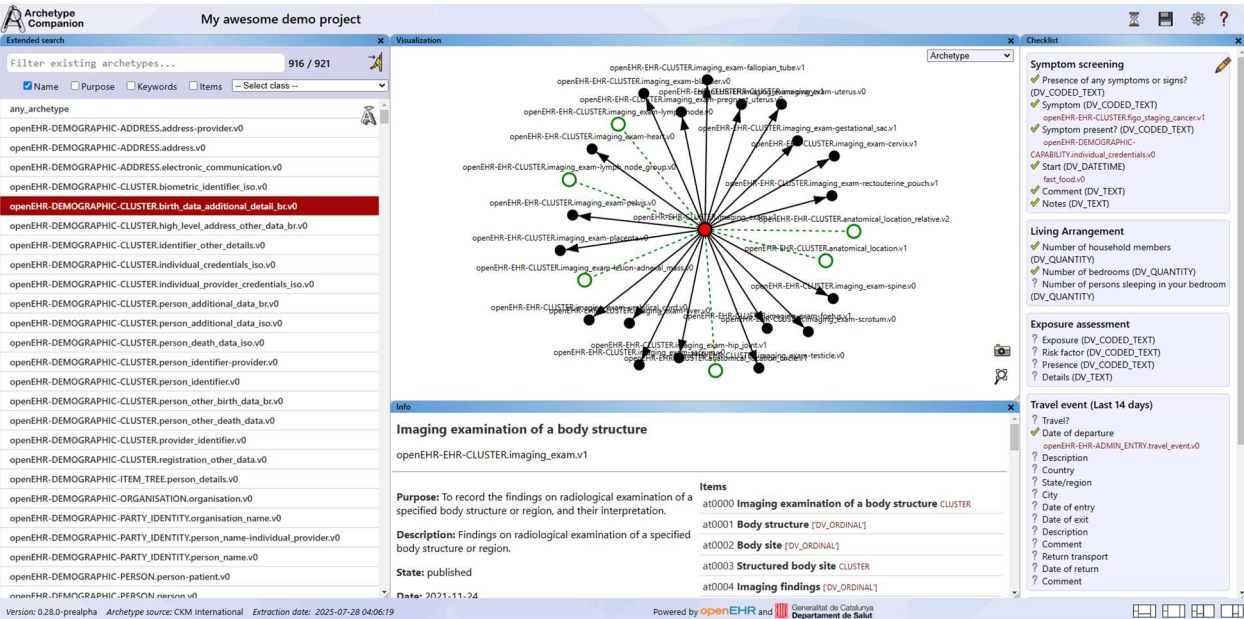
Checklist editor view

The screenshot shows the 'Checklist editor view' of the Archetype Companion. The main window displays the 'Imaging examination of a body structure' archetype. The left sidebar lists various archetypes, including 'openEHR-DEMOGRAPHIC-CLUSTER.person_other_birth_data.v0' and 'openEHR-DEMOGRAPHIC-CLUSTER.person_other_death_data.v0'. The main content area shows the 'Imaging examination of a body structure' archetype with its purpose, description, state, date, keywords, parent, children, and items. The 'Items' list includes 'at0000 Imaging examination of a body structure CLUSTER', 'at0001 Body structure [DV_ORDINAL]', 'at0002 Body site [DV_ORDINAL]', and 'at0003 Structured body site CLUSTER'. The right sidebar shows a 'Checklist editor' with sections for 'Symptom screening', 'Living Arrangement', 'Exposure assessment', and 'Travel event (Last 14 days)'. The bottom status bar indicates 'Version: 0.28.0-prealpha', 'Archetype source: CKM International', 'Extraction date: 2025-07-28 04:06:19', and 'Powered by openEHR and Generalitat de Catalunya Departament de Salut'.

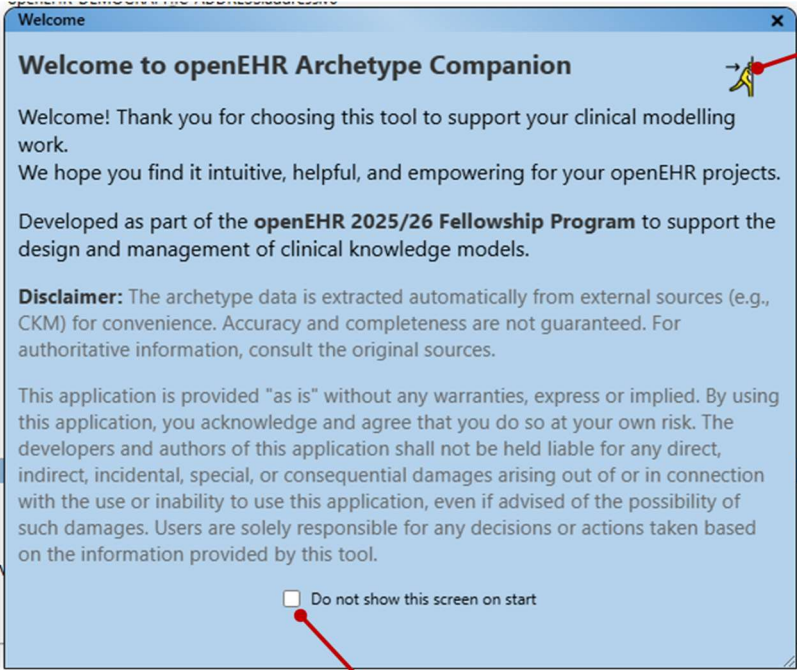
The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

Martin Andreas Koch

Search view



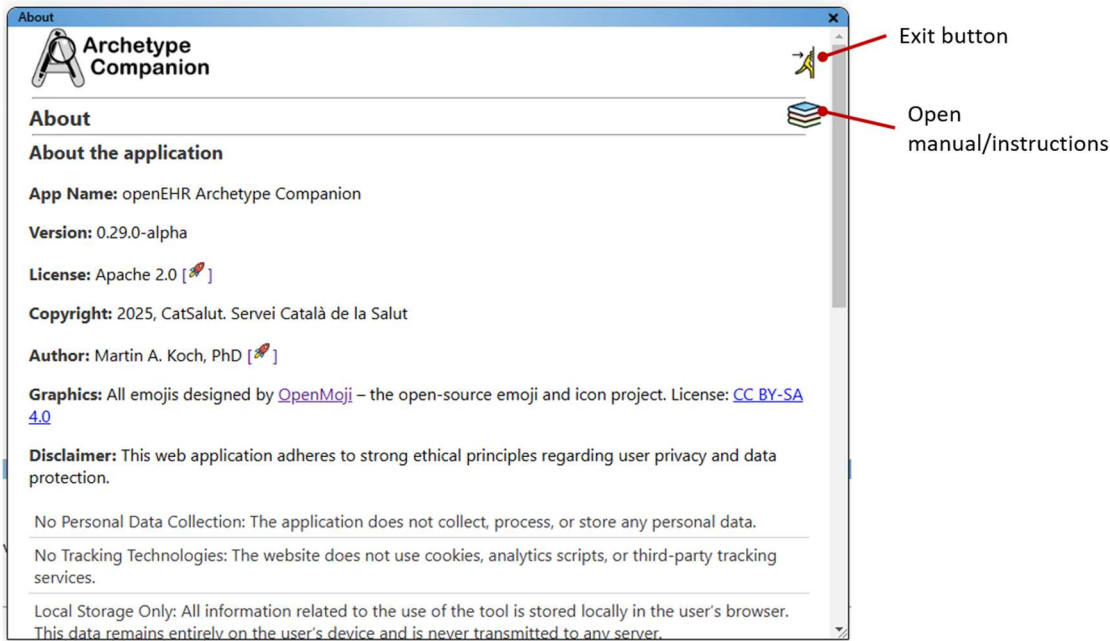
Splash Screen



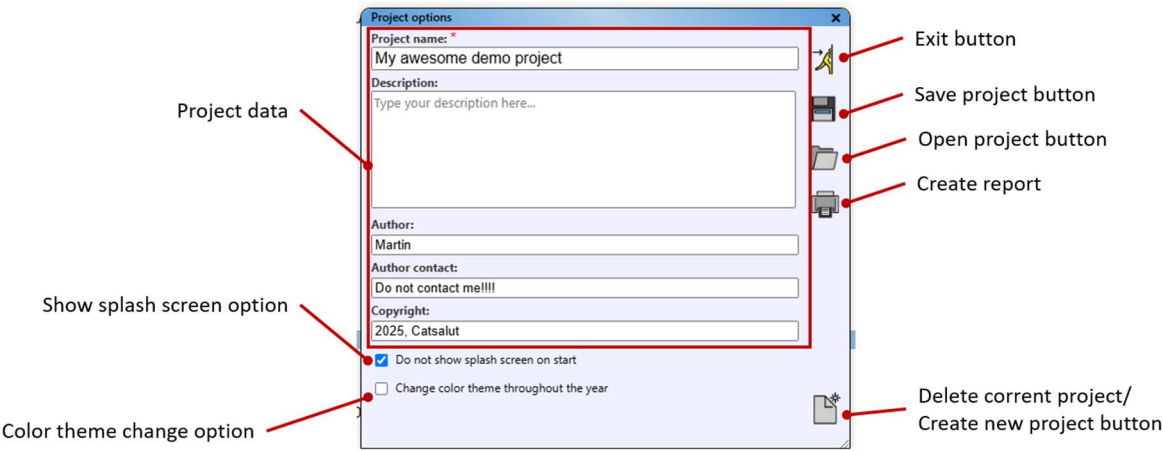
Exit button

Do not show on startup option

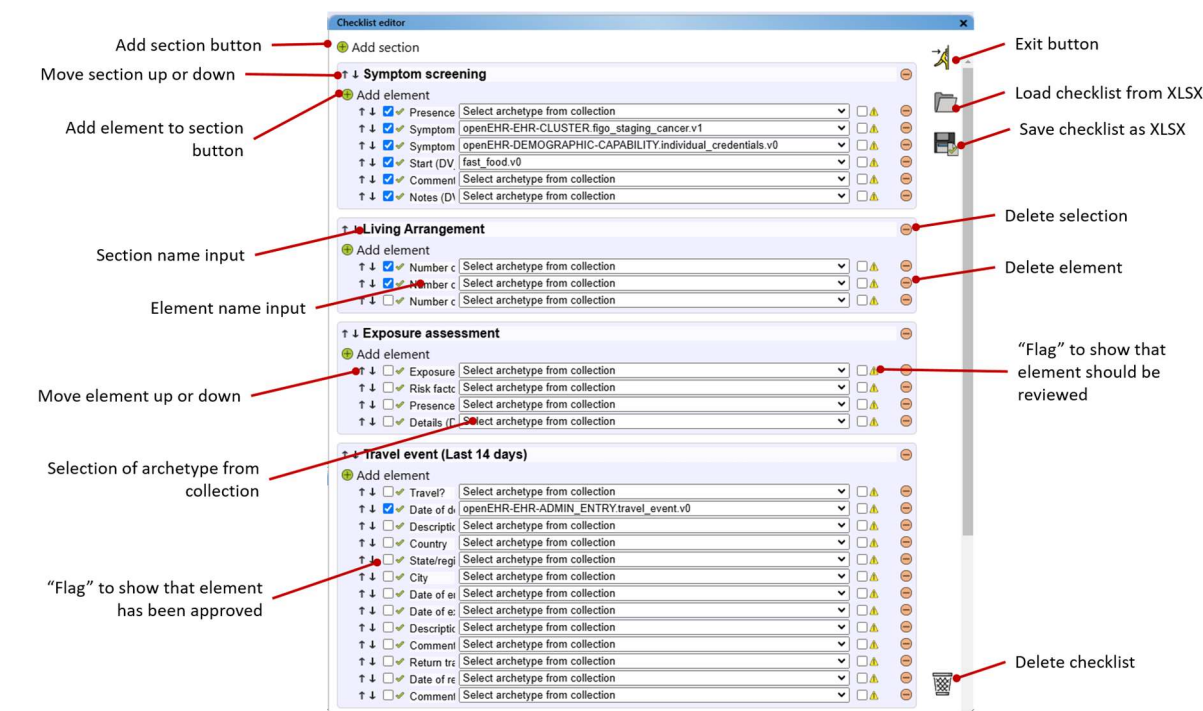
About Page



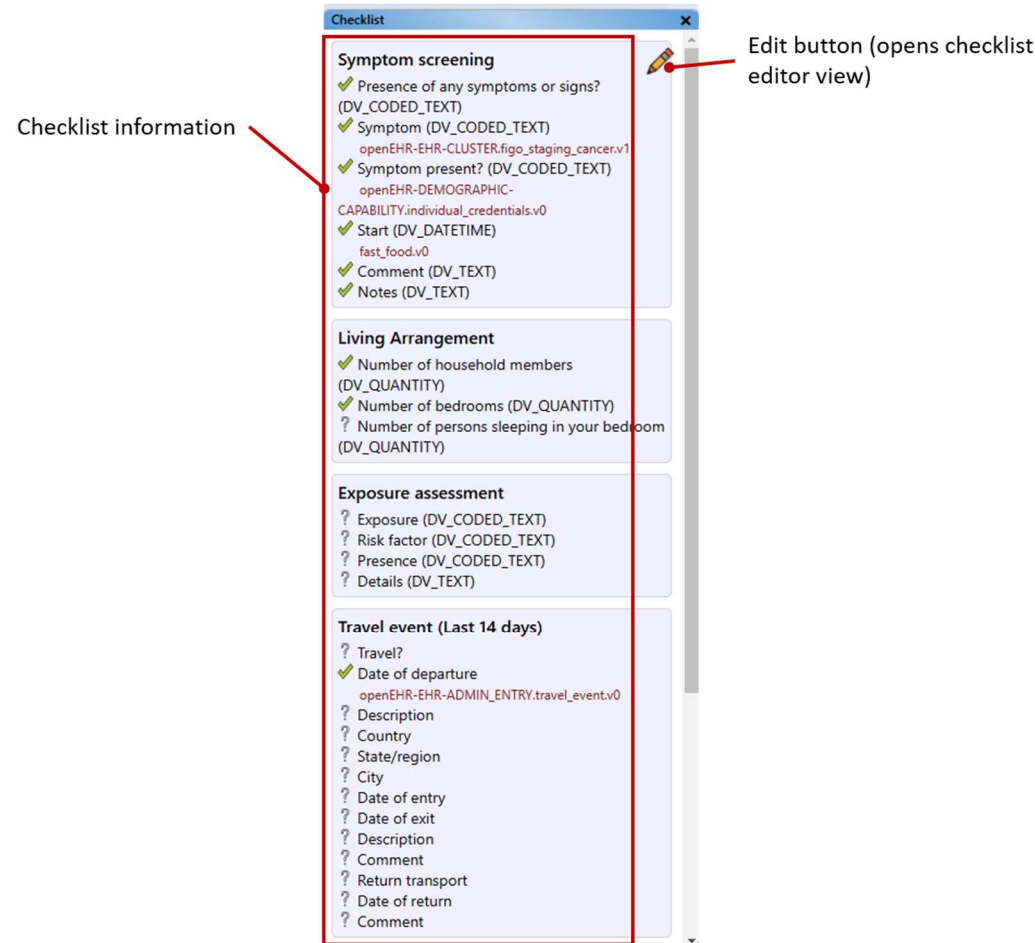
Project Options View



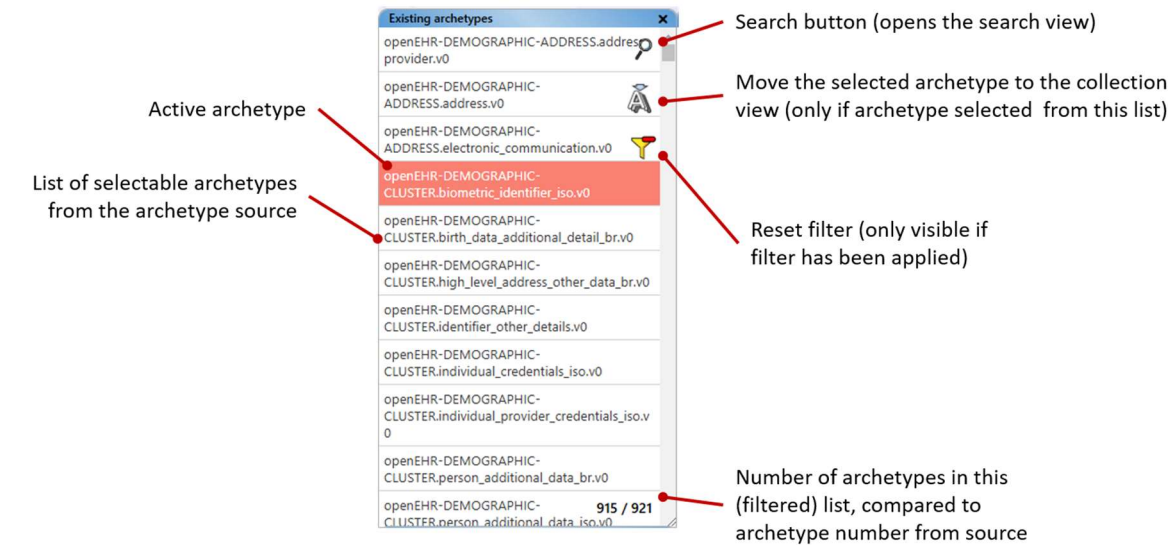
Checklist Editor View



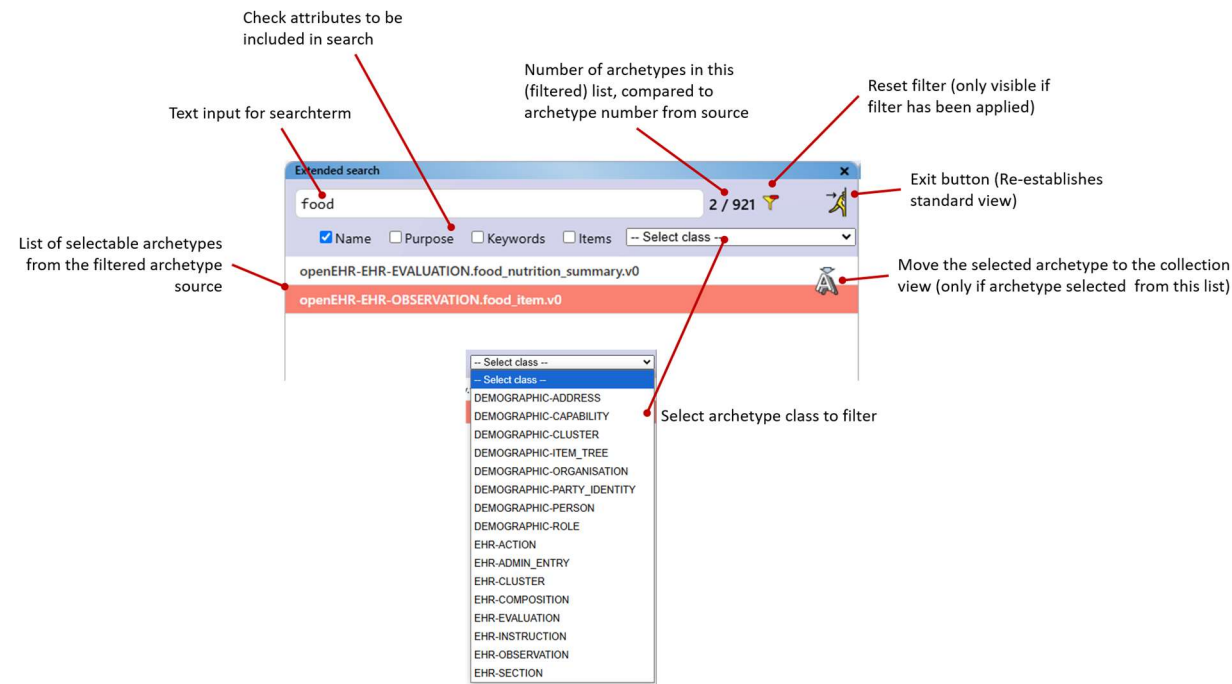
Checklist Sidebar



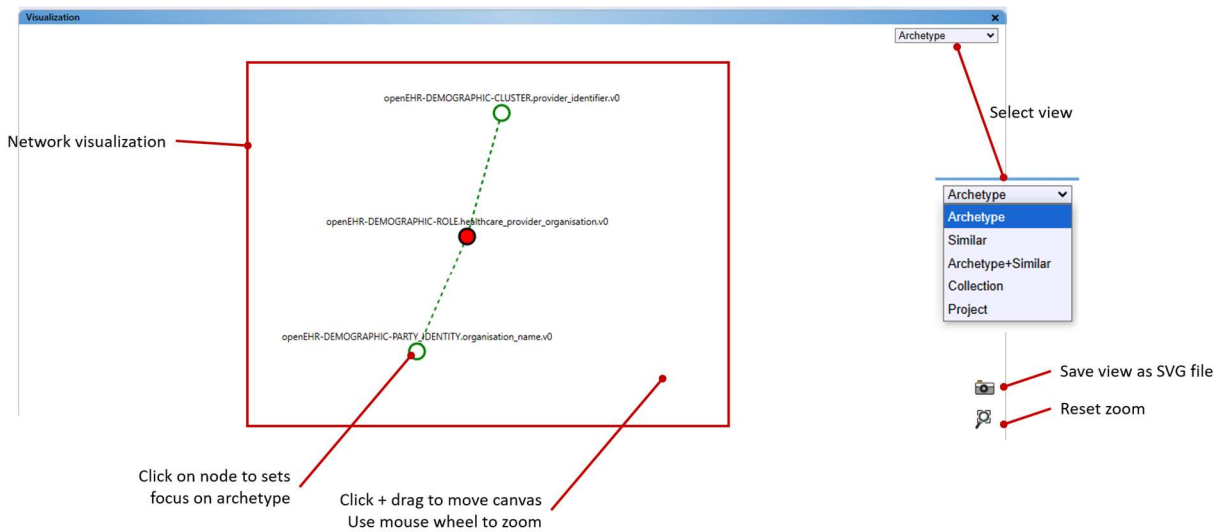
Archetype List Sidebar



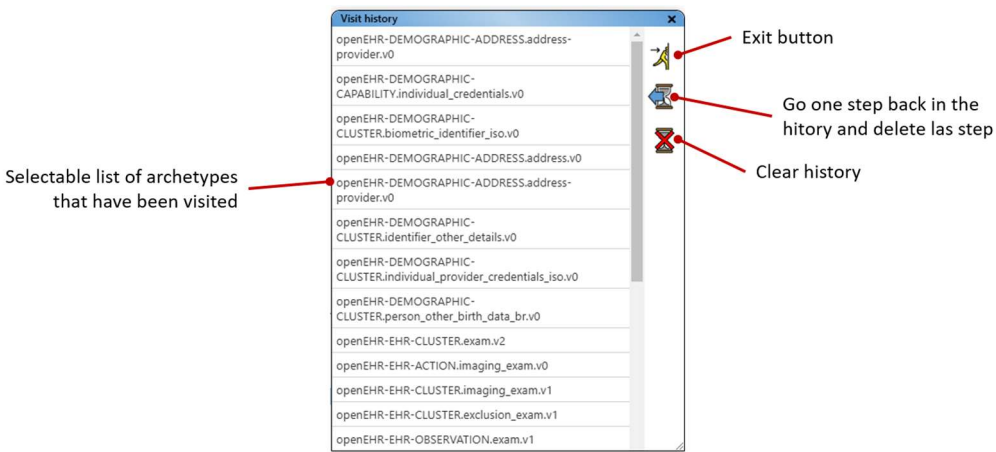
Search View



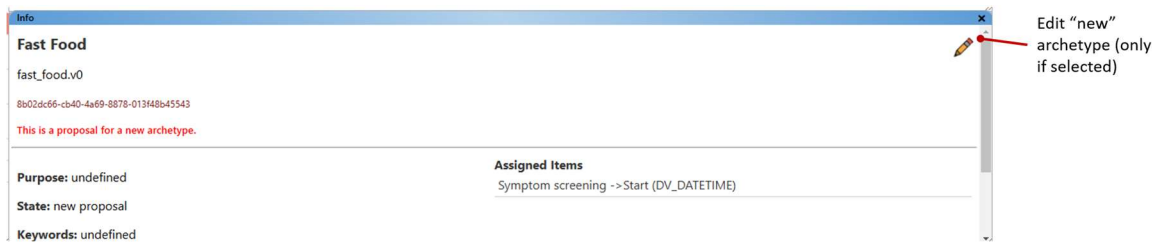
Visualization View



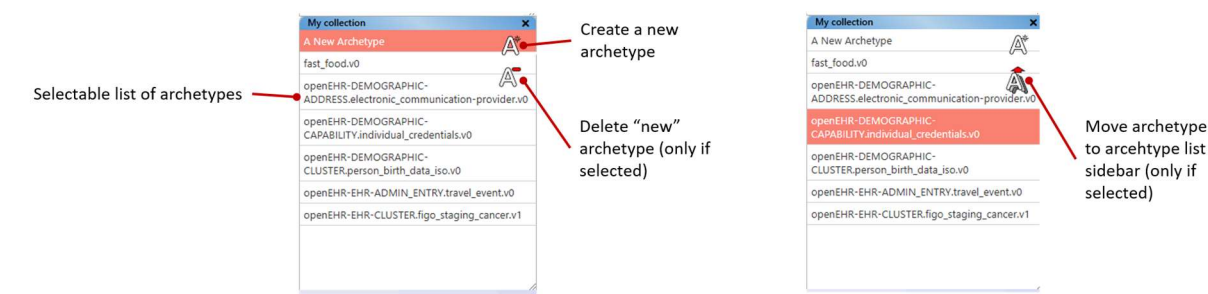
History



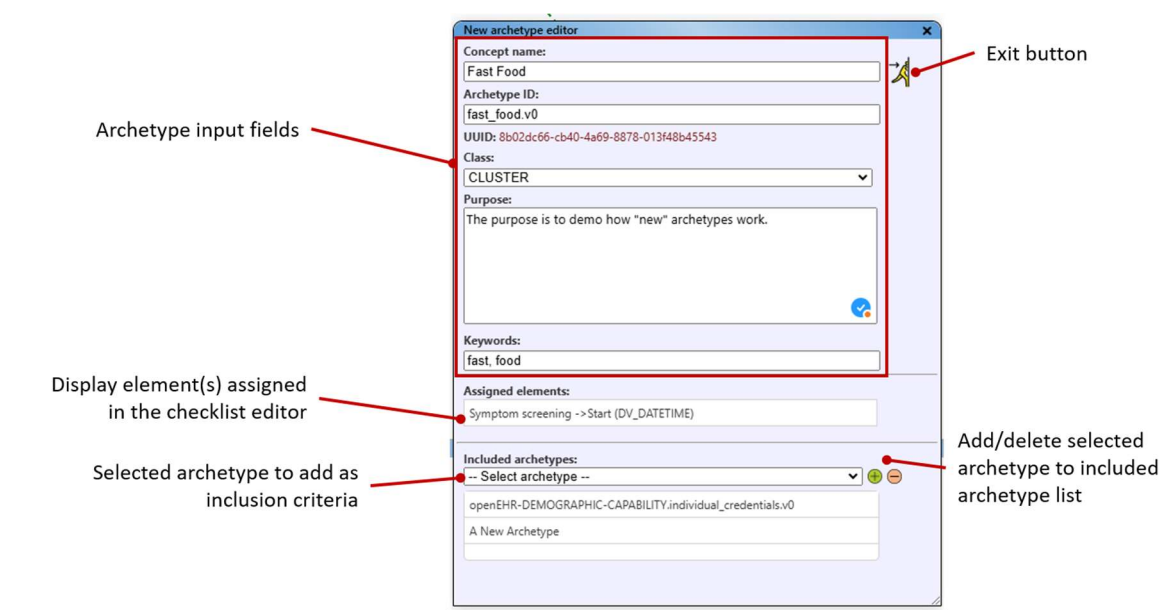
Info View



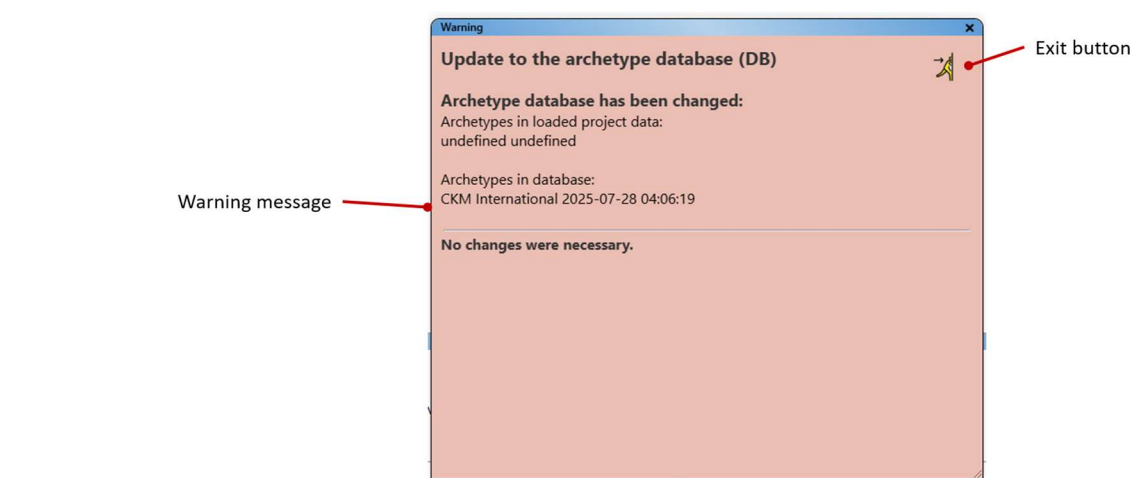
Collection View



New Archetype Editor View

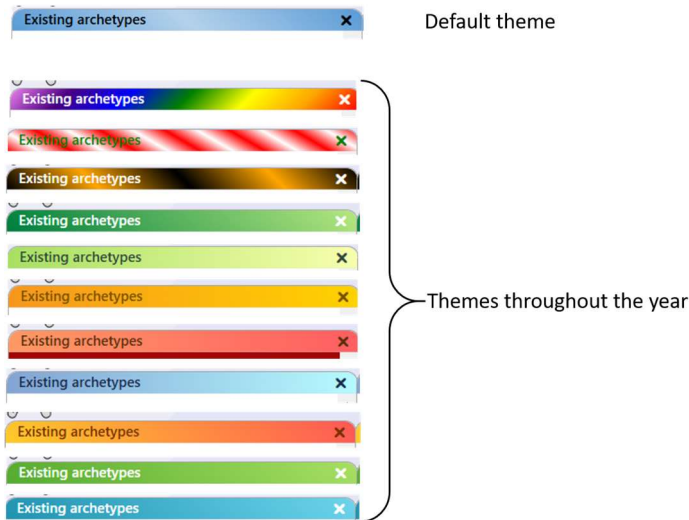


Warning page



Colour themes

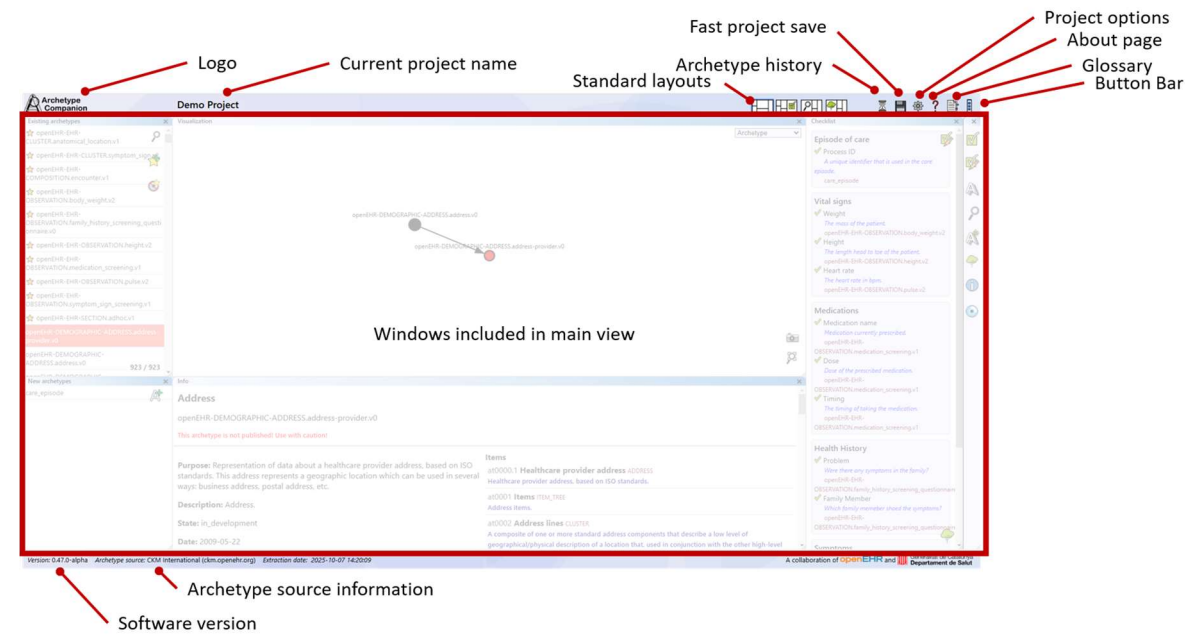
If enabled, the colour theme of the GUI bars is changed depending on the time of year. The different themes are depicted in the following figure.



Third Iteration of the GUI

Using the GUI has shown the potential for improvement. The main additions to this third iteration is the Template Palnner and the openEHR Class Glossary.

Main View



The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

Martin Andreas Koch

Standard view

Archetype Companion Demo Project

Existing archetypes

- openEHR-EHR-CLUSTER.anatomical_location.v1
- openEHR-EHR-CLUSTER.symptom_sign.v1
- openEHR-EHR-COMPOSITION.encounter.v1
- openEHR-EHR-OBSERVATION.body_weight.v2
- openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v1
- openEHR-EHR-OBSERVATION.height.v2
- openEHR-EHR-OBSERVATION.medication_screening.v1
- openEHR-EHR-OBSERVATION.pulse.v2
- openEHR-EHR-OBSERVATION.symptom_sign_screening.v1
- openEHR-EHR-SECTION.adhoc.v1
- openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0**
- openEHR-DEMOGRAPHIC-ADDRESS.address.v0
- care_episode

Visualization

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

openEHR-DEMOGRAPHIC-ADDRESS.address.v0

Info

Address

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

This archetype is not published! Use with caution!

Purpose: Representation of data about a healthcare provider address, based on ISO standards. This address represents a geographic location which can be used in several ways: business address, postal address, etc.

Description: Address.

State: in_development

Date: 2009-05-22

Items

- at0000.1 Healthcare provider address ADDRESS Healthcare provider address, based on ISO standards.
- at0001 Items ITEM_TREE Address items.
- at0002 Address lines CLUSTER A composite of one or more standard address components that describe a low level of geographical/physical description of a location that, used in conjunction with the other high-level

Checklist

Episode of care

- Process ID A unique identifier that is used in the care episode. care_episode

Vital signs

- Weight The mass of the patient. openEHR-EHR-OBSERVATION.body_weight.v2
- Height The length head to toe of the patient. openEHR-EHR-OBSERVATION.height.v2
- Heart rate The heart rate in bpm. openEHR-EHR-OBSERVATION.pulse.v2

Medications

- Medication name Medication currently prescribed. openEHR-EHR-OBSERVATION.medication_screening.v1
- Dose Dose of the prescribed medication. openEHR-EHR-OBSERVATION.medication_screening.v1
- Timing The timing of taking the medication. openEHR-EHR-OBSERVATION.medication_screening.v1

Health History

- Problem Were there any symptoms in the family? openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v1
- Family Member Which family member shared the symptoms? openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v1

Version: 0.47.0-alpha Archetype source: CKM International (ckm.openehr.org) Extraction date: 2025-10-07 14:20:09

Checklist editor view

Archetype Companion Demo Project

Existing archetypes

- openEHR-EHR-CLUSTER.anatomical_location.v1
- openEHR-EHR-CLUSTER.symptom_sign.v1
- openEHR-EHR-COMPOSITION.encounter.v1
- openEHR-EHR-OBSERVATION.body_weight.v2
- openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v1
- openEHR-EHR-OBSERVATION.height.v2
- openEHR-EHR-OBSERVATION.medication_screening.v1
- openEHR-EHR-OBSERVATION.pulse.v2
- openEHR-EHR-OBSERVATION.symptom_sign_screening.v1
- openEHR-EHR-SECTION.adhoc.v1
- openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0**
- openEHR-DEMOGRAPHIC-ADDRESS.address.v0
- care_episode

Visualization

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

openEHR-DEMOGRAPHIC-ADDRESS.address.v0

Info

Address

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

This archetype is not published! Use with caution!

Purpose: Representation of data about a healthcare provider address, based on ISO standards. This address represents a geographic location which can be used in several ways: business address, postal address, etc.

Description: Address.

State: in_development

Items

- at0000.1 Healthcare provider address ADDRESS Healthcare provider address, based on ISO standards.
- at0001 Items ITEM_TREE Address items.
- at0002 Address lines CLUSTER A composite of one or more standard address components that describe a low level of geographical/physical description of a

Checklist editor

Episode of care

- Add section
- Add element
- Process ID care_episode A unique identifier that is used in the care episode.

Vital signs

- Add element
- Weight openEHR-EHR-OBSERVATION.body_weight.v2 The mass of the patient.
- Height openEHR-EHR-OBSERVATION.height.v2 The length head to toe of the patient.
- Heart rate openEHR-EHR-OBSERVATION.pulse.v2 The heart rate in bpm.

Medications

- Add element
- Medication name openEHR-EHR-OBSERVATION.medication_screening.v1 Medication currently prescribed.
- Dose openEHR-EHR-OBSERVATION.medication_screening.v1 Dose of the prescribed medication.
- Timing openEHR-EHR-OBSERVATION.medication_screening.v1 The timing of taking the medication.

Health History

- Add element

Version: 0.47.0-alpha Archetype source: CKM International (ckm.openehr.org) Extraction date: 2025-10-07 14:20:09

The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

Martin Andreas Koch

Search view

Archetype Companion Demo Project

Search: Filter existing archetypes... 923 / 923

Archetype: openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

Visualization: openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

Info: Address

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

This archetype is not published! Use with caution!

Purpose: Representation of data about a healthcare provider address, based on ISO standards. This address represents a geographic location which can be used in several ways: business address, postal address, etc.

Description: Address.

State: in_development

Items:

- at0000.1 Healthcare provider address ADDRESS
- Healthcare provider address, based on ISO standards.
- at0001 Items ITEM_TREE
- Address Items.
- at0002 Address lines CLUSTER
- A composite of one or more standard address components that describe a low level of geographical/physical description of a location that, used in

Checklist:

- Episode of care
 - Process ID
 - A unique identifier that is used in the care episode
 - care_episode
- Vital signs
 - Weight
 - The mass of the patient
 - openEHR-EHR-OBSERVATION.body_weight.v2
 - Height
 - The length head to toe of the patient
 - openEHR-EHR-OBSERVATION.height.v2
 - Heart rate
 - The heart rate in bpm
 - openEHR-EHR-OBSERVATION.pulse.v2
- Medications
 - Medication name
 - Medication currently prescribed
 - openEHR-EHR-OBSERVATION.medication_screening.v1
 - Dose
 - Dose of the prescribed medication
 - openEHR-EHR-OBSERVATION.medication_screening.v1
 - Timing
 - The timing of taking the medication
 - openEHR-EHR-OBSERVATION.medication_screening.v1
- Health History
 - Problem
 - Were there any symptoms in the family?
 - openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v1
 - Family Member
 - Which family member should the symptoms?
 - openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v1

Version: 0.47.0-alpha Archetype source: CKM International (ckm.openehr.org) Extraction date: 2025-10-07 14:20:09

A collaboration of openEHR and Generalitat de Catalunya Departament de Salut

Template Planner view

Archetype Companion Demo Project

Template Planner: openEHR-EHR-COMPOSITION encounter.v1

Visualization: openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

Info: Address

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

This archetype is not published! Use with caution!

Purpose: Representation of data about a healthcare provider address, based on ISO standards. This address represents a geographic location which can be used in several ways: business address, postal address, etc.

Description: Address.

Items:

- at0000.1 Healthcare provider address ADDRESS
- Healthcare provider address, based on ISO standards.
- at0001 Items ITEM_TREE
- Address Items.
- at0002 Address lines CLUSTER
- A composite of one or more standard address components that describe a low level of geographical/physical description of a location that, used in

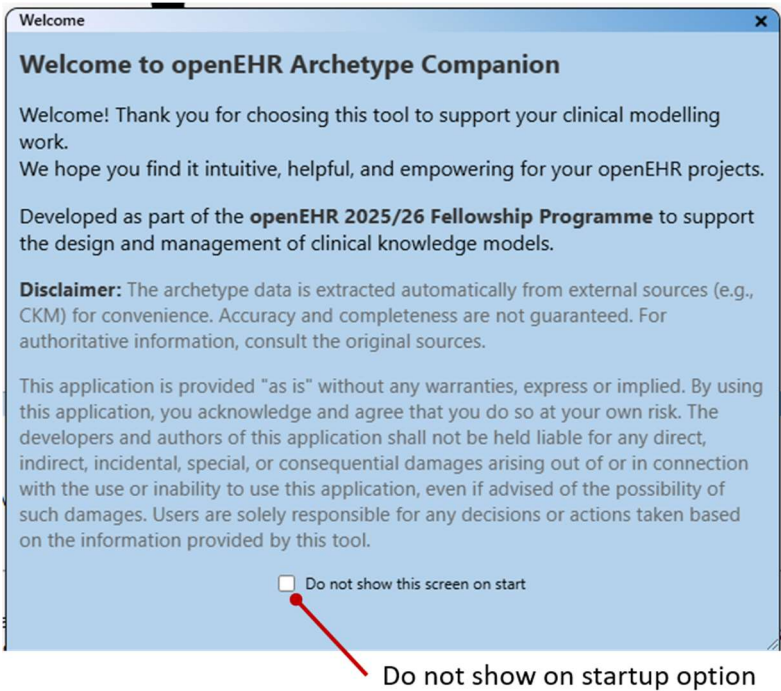
Checklist:

- Episode of care
 - Process ID
 - A unique identifier that is used in the care episode
 - care_episode
- Vital signs
 - Weight
 - The mass of the patient
 - openEHR-EHR-OBSERVATION.body_weight.v2
 - Height
 - The length head to toe of the patient
 - openEHR-EHR-OBSERVATION.height.v2
 - Heart rate
 - The heart rate in bpm
 - openEHR-EHR-OBSERVATION.pulse.v2
- Medications
 - Medication name
 - Medication currently prescribed
 - openEHR-EHR-OBSERVATION.medication_screening.v1
 - Dose
 - Dose of the prescribed medication
 - openEHR-EHR-OBSERVATION.medication_screening.v1
 - Timing
 - The timing of taking the medication
 - openEHR-EHR-OBSERVATION.medication_screening.v1
- Health History
 - Problem
 - Were there any symptoms in the family?
 - openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v1
 - Family Member
 - Which family member should the symptoms?
 - openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v1

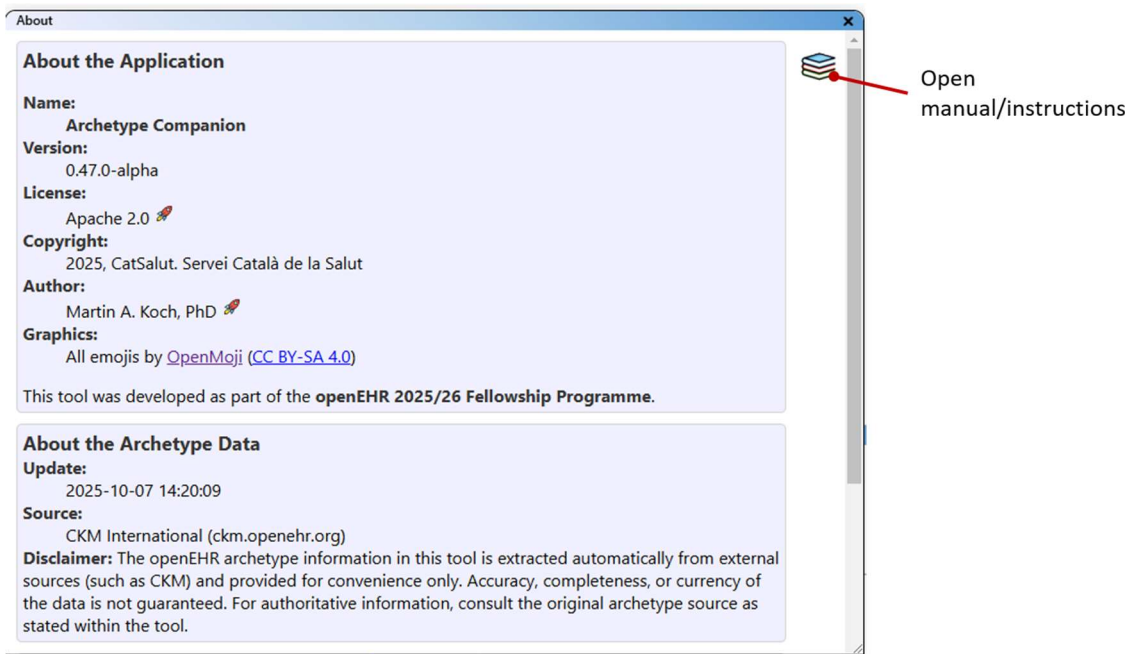
Version: 0.47.0-alpha Archetype source: CKM International (ckm.openehr.org) Extraction date: 2025-10-07 14:20:09

A collaboration of openEHR and Generalitat de Catalunya Departament de Salut

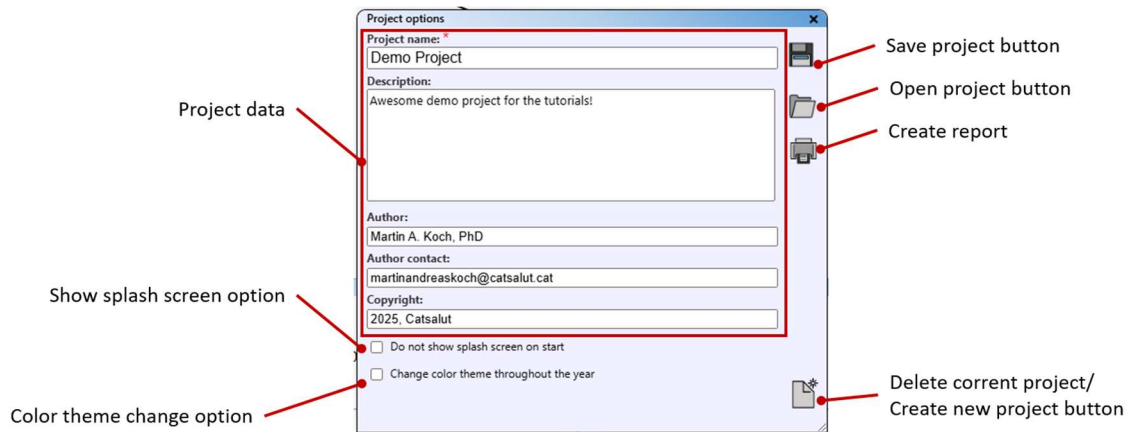
Splash Screen



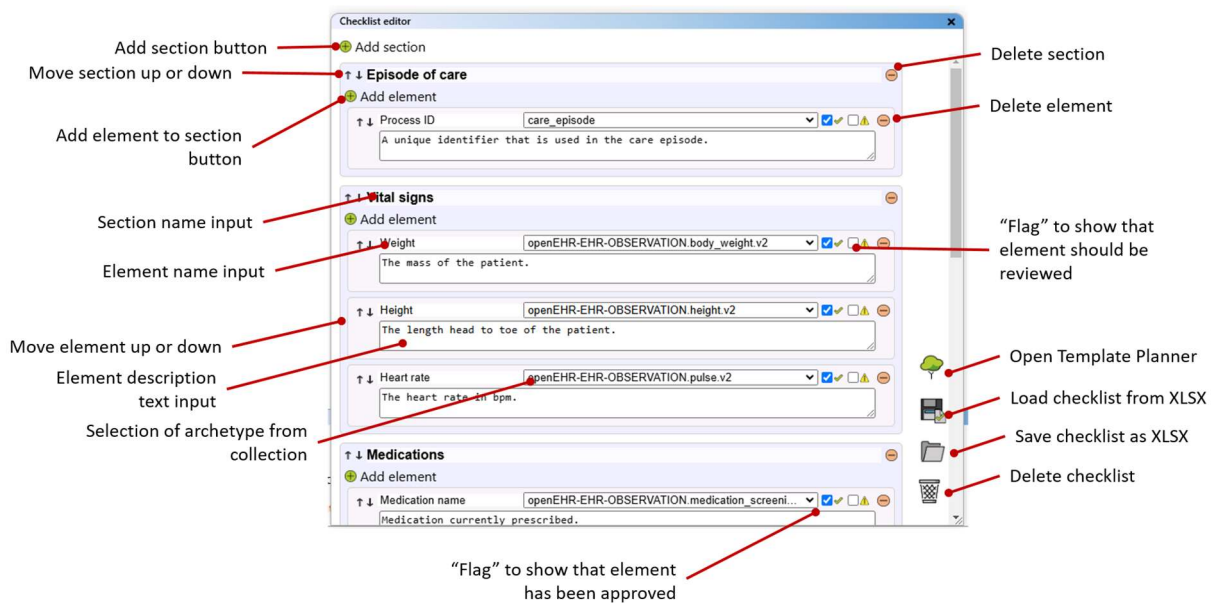
About Page



Project Options View



Checklist Editor View



Checklist

Episode of care

- ✓ Process ID
A unique identifier that is used in the care episode.
care_episode

Vital signs

- ✓ Weight
The mass of the patient.
openEHR-EHR-OBSERVATION.body_weight.v2
- ✓ Height
The length head to toe of the patient.
openEHR-EHR-OBSERVATION.height.v2
- ✓ Heart rate
The heart rate in bpm.
openEHR-EHR-OBSERVATION.pulse.v2

Medications

- ✓ Medication name
Medication currently prescribed.
openEHR-EHR-OBSERVATION.medication_screening.v1
- ✓ Dose
Dose of the prescribed medication.
openEHR-EHR-OBSERVATION.medication_screening.v1
- ✓ Timing
The timing of taking the medication.
openEHR-EHR-OBSERVATION.medication_screening.v1

Health History

- ✓ Problem
Were there any symptoms in the family?
openEHR-EHR-OBSERVATION.family_history_screening_questionnaire
- ✓ Family Member
Which family member should the symptoms?
openEHR-EHR-OBSERVATION.family_history_screening_questionnaire

Symptoms

Edit button (opens checklist editor view)

Open Template Planner

Existing archetypes

- openEHR-EHR-CLUSTER.anatomical_location.v1
- openEHR-EHR-CLUSTER.symptom_sign.v1
- openEHR-EHR-COMPOSITION.encounter.v1
- openEHR-EHR-OBSERVATION.body_weight.v2
- openEHR-EHR-OBSERVATION.family_history_screening_questionnaire.v0
- openEHR-EHR-OBSERVATION.height.v2
- openEHR-EHR-OBSERVATION.medication_screening.v1
- openEHR-EHR-OBSERVATION.pulse.v2
- openEHR-EHR-OBSERVATION.symptom_sign_screening.v1
- openEHR-EHR-SECTION.adhoc.v1
- openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0
- openEHR-DEMOGRAPHIC-ADDRESS.address.v0

923 / 923

Existing archetypes

- openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0
- openEHR-DEMOGRAPHIC-ADDRESS.address.v0
- openEHR-DEMOGRAPHIC-ADDRESS.electronic_communication-provider.v0
- openEHR-DEMOGRAPHIC-ADDRESS.electronic_communication.v0
- openEHR-DEMOGRAPHIC-CAPABILITY.individual_credentials.v0
- openEHR-DEMOGRAPHIC-CLUSTER.biometric_identifier_iso.v0
- openEHR-DEMOGRAPHIC-CLUSTER.birth_data_additional_detail_br.v0
- openEHR-DEMOGRAPHIC-CLUSTER.high_level_address_other_data_br.v0
- openEHR-DEMOGRAPHIC-CLUSTER.identifier_other_details.v0
- openEHR-DEMOGRAPHIC-CLUSTER.individual_credentials_iso.v0
- openEHR-DEMOGRAPHIC-CLUSTER.individual_provider_credentials_iso.v0

32 / 923

The screenshot shows the Search interface with the following components and annotations:

- Search bar:** Contains the text "openEHR-D".
- Attributes:** Checkboxes for ID (checked), Name, Purpose, Keywords, Items, and Inclusions.
- Class selector:** A dropdown menu labeled "-- Select class --" with a list of selectable archetypes.
- Archetype list:** A list of archetypes, including "openEHR-DEMOGRAPHIC-ADDRESS.provider.v0", "openEHR-DEMOGRAPHIC-ADDRESS.address.v0", and "openEHR-DEMOGRAPHIC-ADDRESS.provider.v0".
- Annotations:**
 - "Text input for searchterm" points to the search bar.
 - "Check attributes to be included in search" points to the attribute checkboxes.
 - "List of selectable archetypes from the filtered archetype source" points to the class selector dropdown.
 - "Number of archetypes in this (filtered) list, compared to archetype number from source" points to the "32 / 923" indicator.
 - "Reset filter (only visible if filter has been applied)" points to the reset button (circular arrow icon).
 - "Add/remove the selected archetype to the collection view (only if archetype selected from this list)" points to the star icon.
 - "Hide/show 'selection' archetypes" points to the magnifying glass icon.
 - "Select archetype class to filter" points to the class selector dropdown.

The screenshot shows a software interface for visualizing a network. The main window, titled 'Visualization', contains a network diagram with three nodes connected by dashed green lines. The nodes are labeled: 'openEHR-DEMOGRAPHIC-CLUSTER.provider_identifier.v0' (top, green circle), 'openEHR-DEMOGRAPHIC-ROLE.healthcare_provider_organisation.v0' (middle, red circle), and 'openEHR-DEMOGRAPHIC-PARTY_IDENTITY.organisation_name.v0' (bottom, green circle). A red box highlights the network diagram area. To the right of the main window is a sidebar with several controls: a 'Select view' dropdown menu currently showing 'Archetype', a 'Save view as SVG file' button with a camera icon, and a 'Reset zoom' button with a magnifying glass icon. Red arrows point from text labels to these controls. At the bottom of the main window, there are three red arrows pointing to the network diagram with associated text labels: 'Click on node to sets focus on archetype', 'Click + drag to move canvas', and 'Use mouse wheel to zoom'.

Visualization

Network visualization

Click on node to sets focus on archetype

Click + drag to move canvas

Use mouse wheel to zoom

Archetype

Select view

Archetype

Archetype

Similar

Archetype+Similar

Collection

Project

Template Planner

Save view as SVG file

Reset zoom

History

Visit history

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

openEHR-EHR-OBSERVATION.medication_screening.v1

openEHR-DEMOGRAPHIC-CLUSTER.biometric_identifier_iso.v0

care_episode

Selectable list of archetypes that have been visited

Go one step back in the history and delete las step

Clear history

Info View

Info

Care Episode

care_episode

61bb534e-2f0b-4cb3-9dcb-a37e50e01163

This is a proposal for a new archetype.

Purpose: A placeholder archetype for my demo project, where I can store a Process ID

Assigned Items

Episode of care - Process ID

State: new_proposal

Keywords: care, episode

Class: CLUSTER

Edit "new" archetype (only if selected)

New Archetype List View

New archetypes

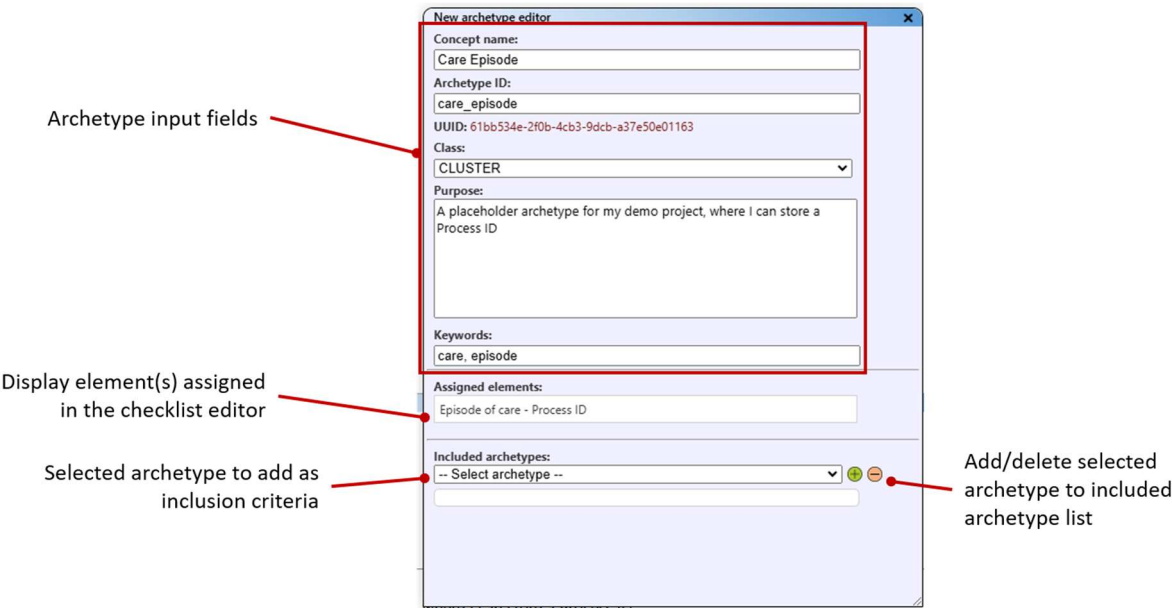
care_episode

Selectable list of "new" archetypes

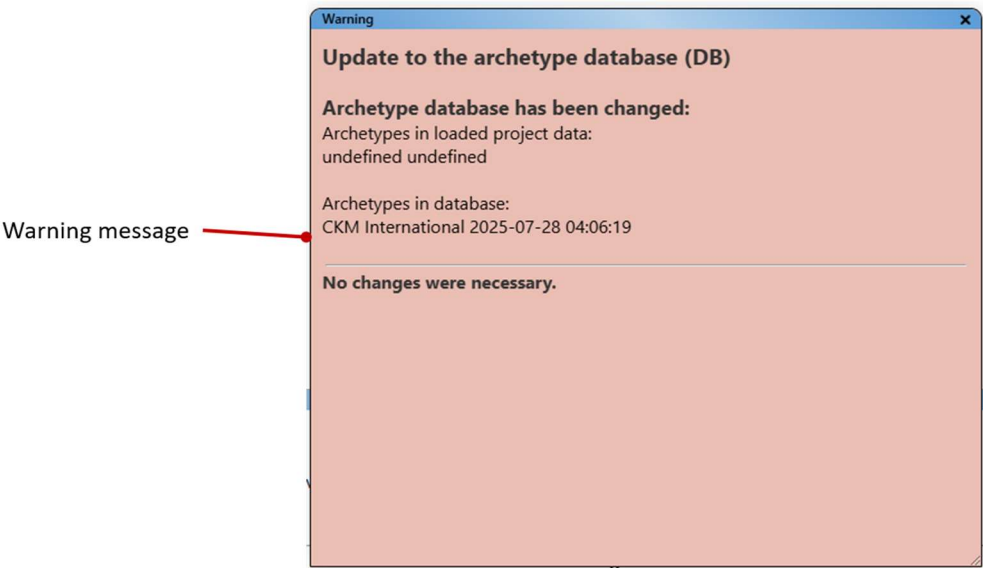
Create a new archetype

Delete "new" archetype (only if selected)

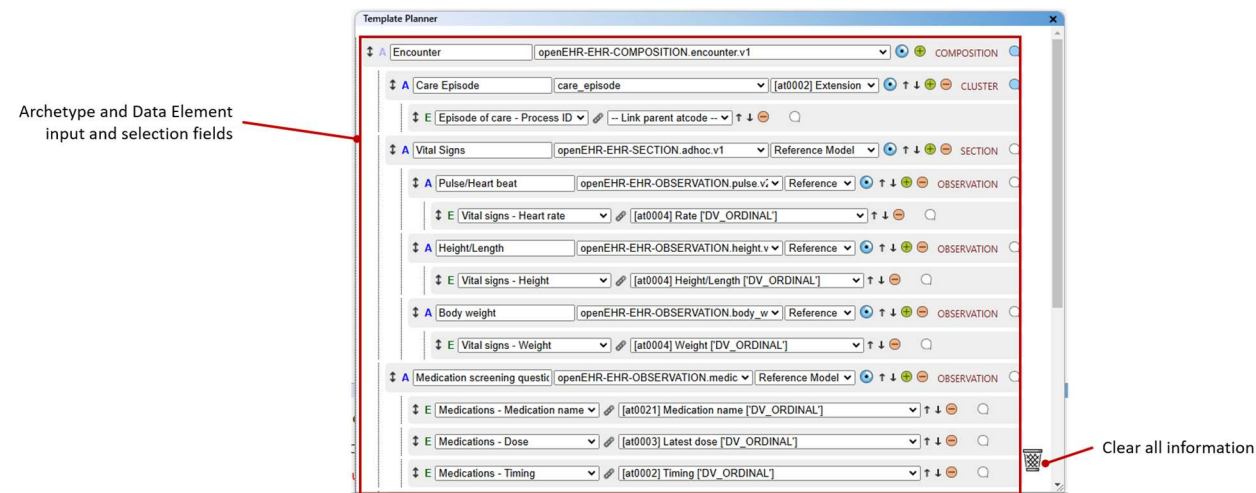
New Archetype Editor View



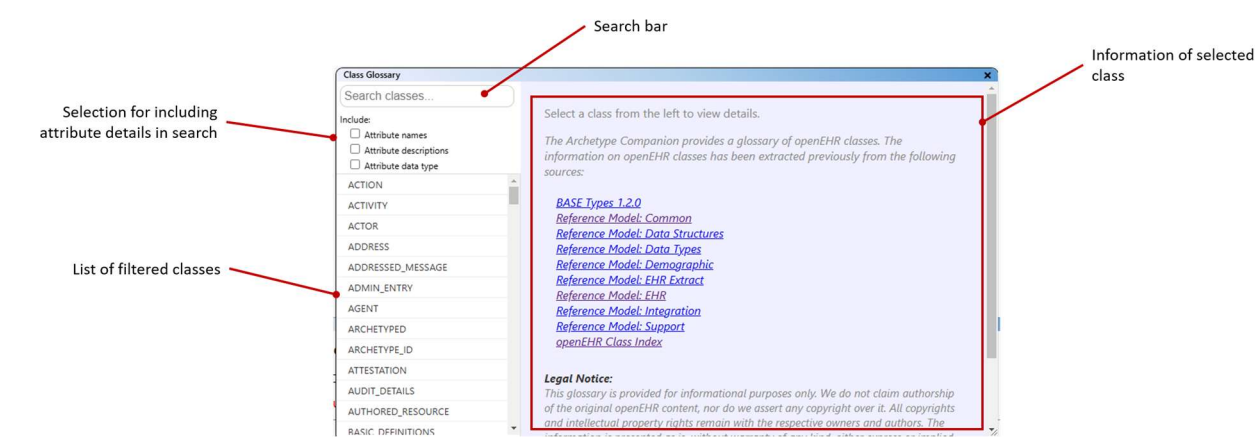
Warning page



Template Planner Editor



openEHR Class Glossary



9.4. Detailed description of function

For a detailed description of the function, please refer to [Annex II – Archetype Designer – User Manual](#).

9.5. How elements and archetypes are connected in the Archetype Companion

In the Archetype Companion, the workflow is based on two main entities:

1. **Data elements** that need to be modelled
2. **A collection of archetypes** that will be used for the model

The goal is to build an **archetype structure** that contains nodes for both archetypes and elements. To support this, **connections between data elements and archetypes** can be defined.

Step 1: Define data elements

First, all relevant **data elements** are added to the **checklist**.

At this stage, they are simply items that represent information to be captured (e.g. blood pressure, diagnosis, date of birth).

Step 2: Create an archetype collection

Next, a **collection of archetypes** is created that is intended for use in the model.

This collection is a curated set of archetypes relevant for the specific project or use case.

Step 3: Link elements to archetypes

After that, in the checklist, **each data element can be linked to an archetype** from the collection (if applicable).

Rules for these links:

- **Each data element can be linked to exactly one archetype.**
(0 or 1 archetype, but not several.)
- **Each archetype can be linked to many data elements.**
- **Not all data elements must have an archetype assigned.**
- **Not all archetypes must have a data element assigned.**

The relationship can be summarized as:

- **data element → at most one archetype**
- **archetype → any number of data elements (including zero)**

These links are later used in the planner to streamline element selection.

Step 4: Use the links in the planner (structure)

In the **planner**, the actual **archetype structure** is created:

1. An **archetype node** is added to the structure.
2. For this node, an archetype from the collection can be selected.
3. When an **element node** is added under an archetype node:
 - all data elements that are linked to the archetype of that node are offered for selection, or
 - the element node can remain unlinked and be labeled freely.

Additional rules in the planner:

- **Archetype nodes are not required to represent an archetype from the collection.**
(Generic or placeholder archetype nodes are possible.)
- **Element nodes are not required to use the label of an assigned data element.**
(Ad hoc element labels are possible.)

Compact summary

Data elements are listed in a checklist and a collection of archetypes is defined. Each data element may be linked to at most one archetype, while each archetype may be linked to many elements or none. In the planner, when an archetype node is associated with an archetype from the collection, all data elements linked to that archetype are available for quick selection as element nodes beneath it. Both archetype nodes and element nodes can also be used without such links, allowing flexibility in the final structure.

9.6. Visualization Language

For the visualization tool, a network representation of connections between different archetypes will be visualized. The views defined so far are:

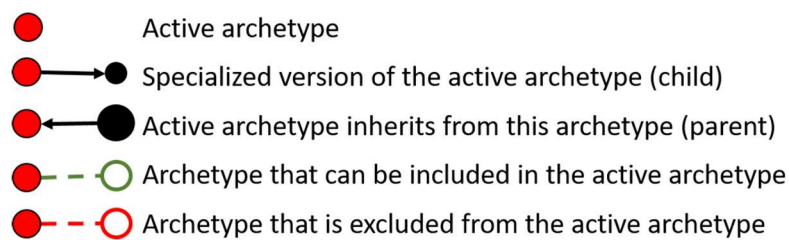
- **Archetype Connection:** For a selected archetype, the relationships to other archetypes should be shown.
 - Parent
 - Children
 - Inclusion
 - Exclusion
- **Similar Archetypes:** For a selected archetype, similar archetypes should be shown.
- **Collection:** For the collection, all archetypes from the collection and their connections should be shown.
- **Project:** For the project, the sections and elements of the checklist should be shown with the corresponding assigned archetypes.

For these four views, a clear and intuitive visual representation has to be defined. In the following sections, a proposal for a representation of every view will be elaborated.

Archetype Connection

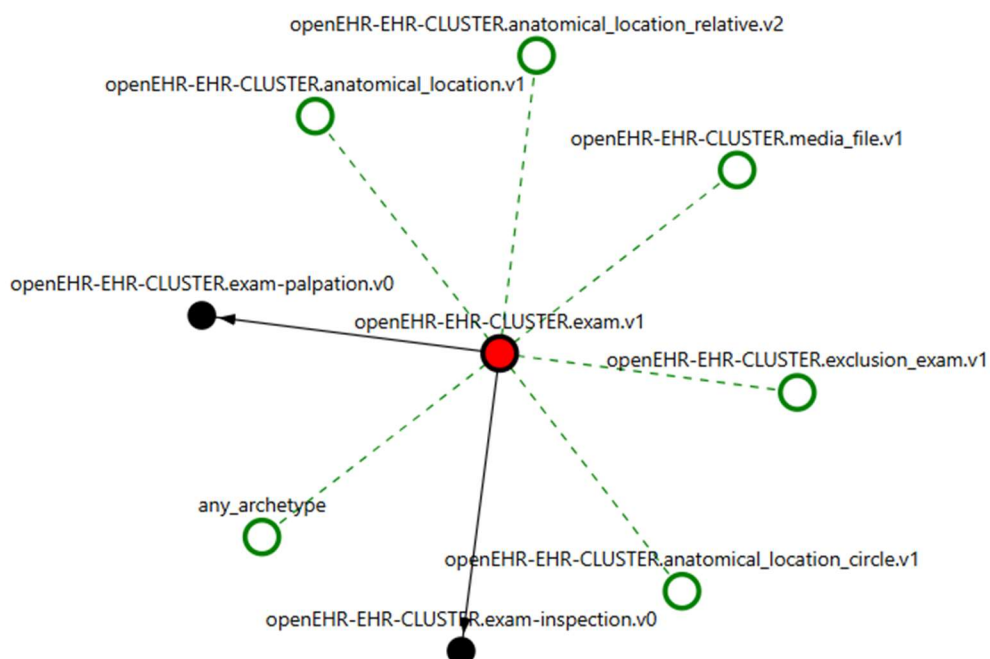
In the **Archetype** view, all relationships between the active archetype and other existing archetypes are visualized. Different types of connections are represented with distinct symbols and line styles for clarity:

- **Specialized archetypes:** Archetypes that are specialized from the active one are shown with a small black disc, an arrow pointing toward the parent, and a solid black line.
- **Parent archetypes:** Archetypes that are parents of the active one are indicated by a large black disc, an arrow pointing toward the parent, and a solid black line.
- **Included archetypes:** Archetypes referenced in inclusion statements within the slots of the active archetype are marked with a green circle and a green dashed connection line.
- **Excluded archetypes:** Archetypes referenced in exclusion statements within the slots of the active archetype are marked with a red circle and a red dashed connection line.



Legend: Symbols and line styles used to represent archetype relationships.

An example of the Archetype view is shown below:

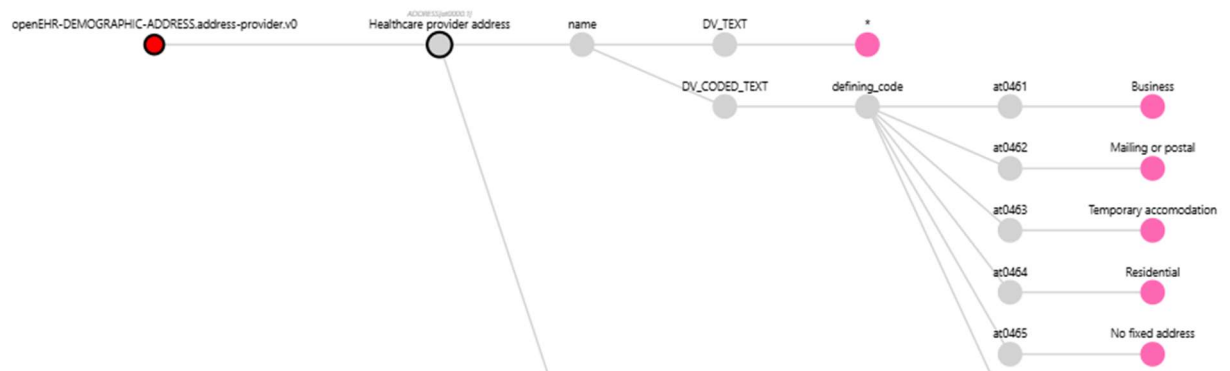


Example visualization of archetype relationships.

Archetype Structure

In the **Archetype Structure** view, a map of the internal structure of the archetype with all elements and defining codes is presented.

Nodes with a black outline have an at-code. Pink nodes are end nodes of the corresponding branch.



Example visualization of archetype structure.

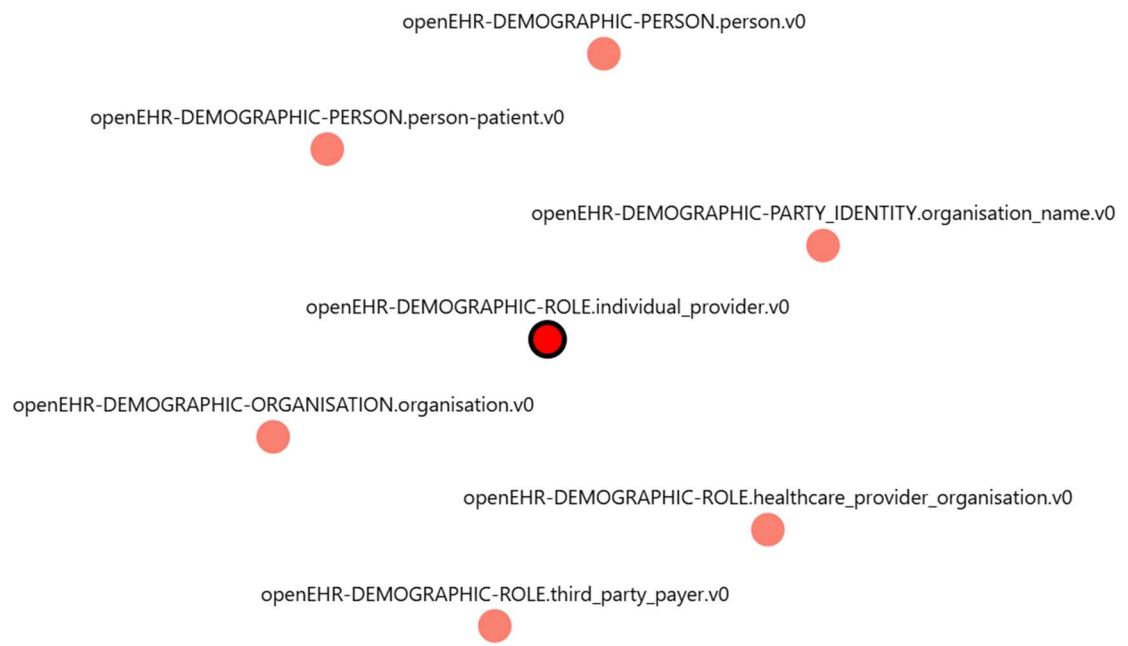
Similar Archetypes

In the **Similar** view, a collection of archetypes that are similar or related to the active archetype is displayed. These similar archetypes are highlighted in a coral color for easy identification. Notably, there are no direct connections shown between the active archetype and the similar ones in this view.

● ● Archetype that is similar to the active archetype

Legend: Similar archetypes are shown in coral red.

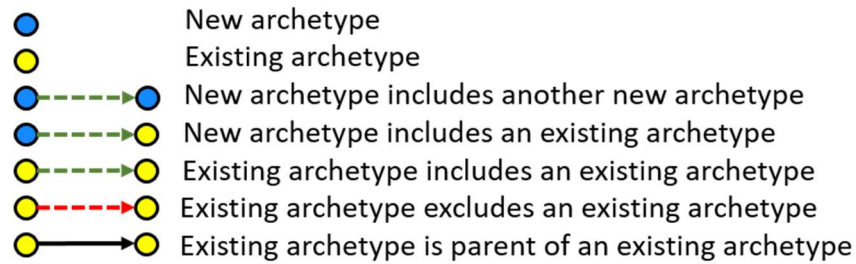
An example of the Similar view is shown below:



Example of similar archetypes displayed in the interface.

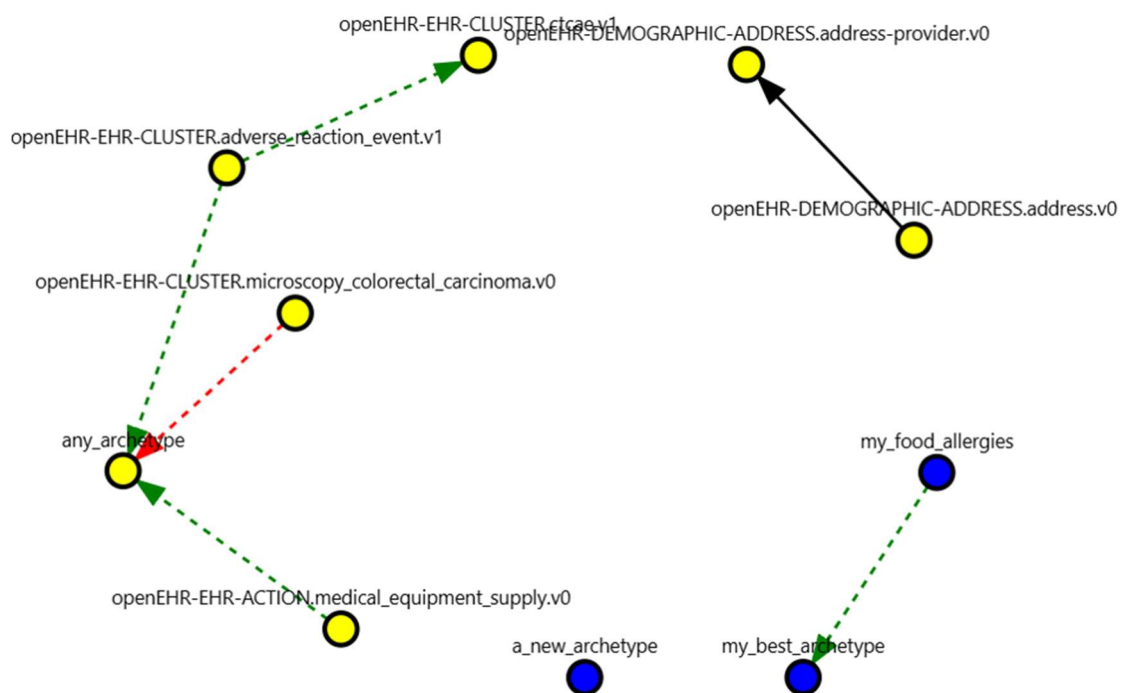
Collection

In the **Collection** view, the "My collection" list displays various archetypes and their relationships. These archetypes include both existing ones from the database and newly proposed archetypes. Relationships can be of the following types: inclusion, exclusion, and parent-child. The coding for these relationships is defined in the following legend:



Legend: Archetype collection colors and connectors.

An example of the Collection view is shown below:



Example of archetype collection displayed in the interface.

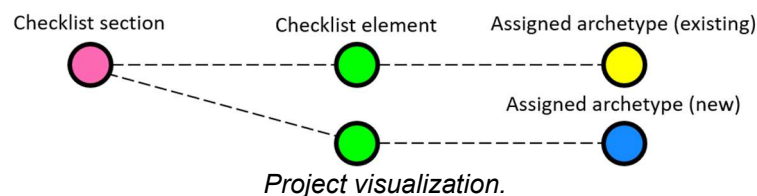
Project

A project is visualized using three columns, arranged from left to right: **Checklist Sections**, **Checklist Elements**, and **Assigned Archetypes**.

Checklist elements are connected to their corresponding sections with a dashed line. When an archetype is assigned to an element, it is displayed in the third column and linked to the element with a dashed line as well.

Color coding of the linked elements:

- **Pink:** Section name
- **Green:** Element name
- **Yellow:** Existing archetype
- **Blue:** New archetype



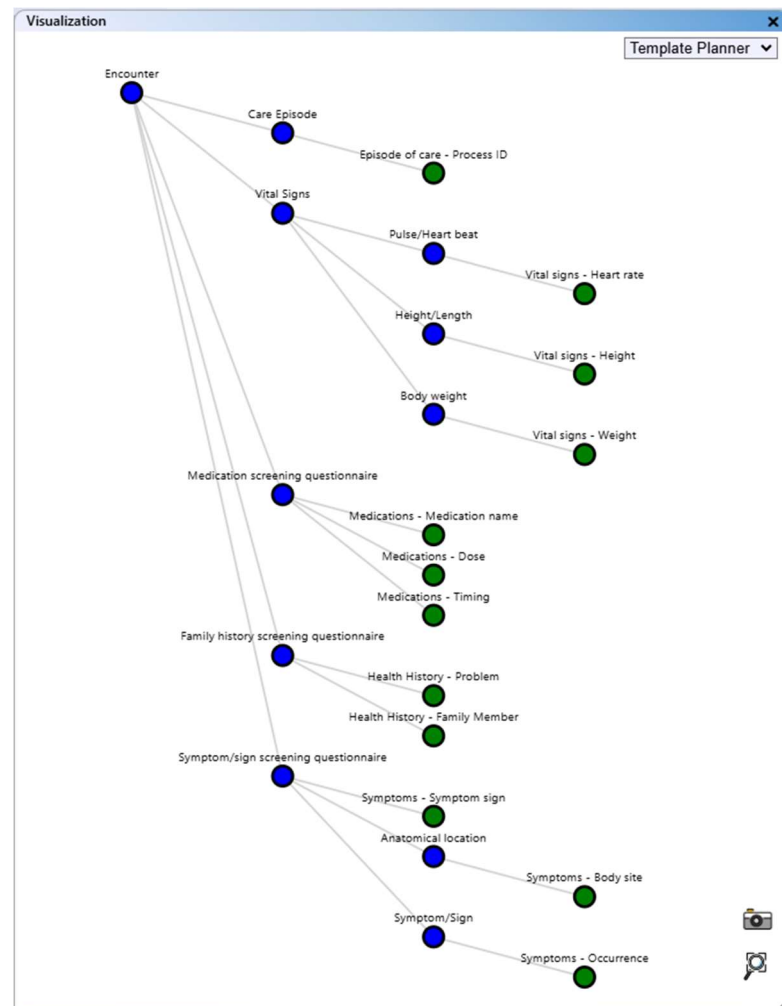
Template planner

In the **Template Planner** view, the tree-like structure of archetypes and data elements is shown.

- Archetype
- Data Element

Legend: Template planner colors.

An example of the Template planner view is shown below:



Example of archetype collection displayed in the interface.

9.7. Local storage

When using the visualization tool, your project data is saved in your browser's **local storage** under the key `ArchetypeExplorerProject`. This ensures your work is not lost when you refresh or close the page*.

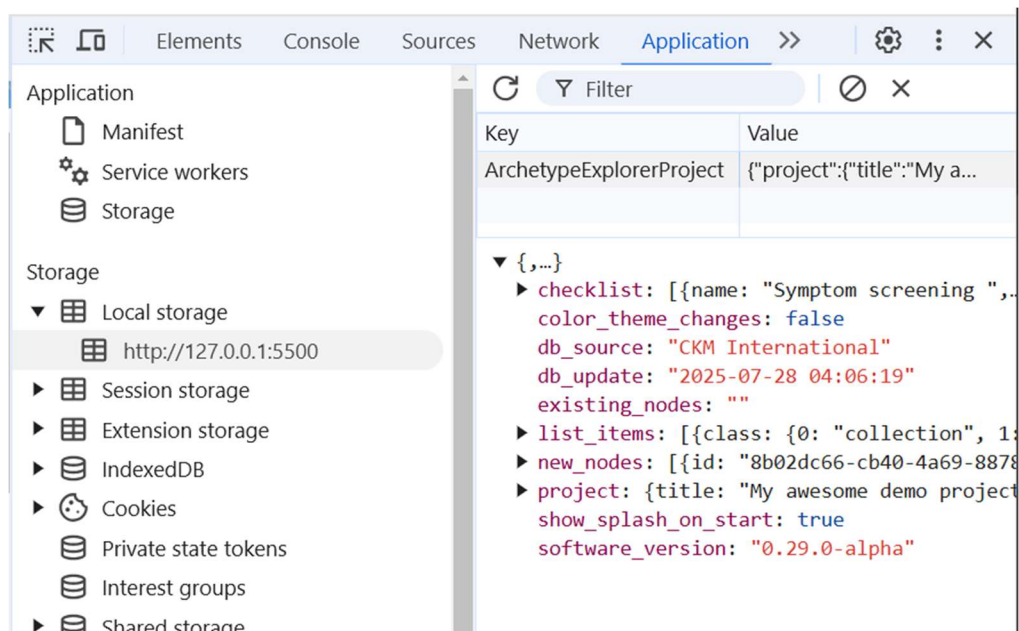
Why This Matters

- **Backup:** Copy the value to save your project elsewhere.
- **Delete:** Remove the variable if you want to reset or clear your data.
- **Restore:** Paste a previously saved value to restore your project.

Here's how you can find, back up, or delete this data in different browsers:

Google Chrome

1. Open Developer Tools: Press Ctrl + Shift + I (Windows/Linux) or Cmd + Option + I (Mac), or right-click the page and select **Inspect**.
2. Go to the **Application** tab.
3. In the left sidebar, expand **Local Storage** and select your website's URL.
4. Find **ArchetypeExplorerProject** in the list. You can view, edit, or delete it by right-clicking the entry.



Mozilla Firefox

1. Open Developer Tools: Press F12 or Ctrl + Shift + I (Windows/Linux) or Cmd + Option + I (Mac), or right-click and select **Inspect**.
2. Click the **Storage** tab.
3. Expand **Local Storage** and select your website.
4. Locate **ArchetypeExplorerProject** in the list. You can view, edit, or delete it as needed.

Microsoft Edge

1. Open Developer Tools: Press F12 or Ctrl + Shift + I (Windows/Linux) or Cmd + Option + I (Mac), or right-click and select **Inspect**.
2. Go to the **Application** tab.
3. Expand **Local Storage** and select your website.
4. Find **ArchetypeExplorerProject** in the list. Right-click to edit or delete.

Safari

1. Enable the Develop menu: Go to **Safari > Preferences > Advanced** and check **Show Develop menu in menu bar**.
 2. Open Web Inspector: Press Cmd + Option + I or select **Develop > Show Web Inspector**.
 3. Click the **Storage** tab.
 4. Expand **Local Storage** and select your website.
 5. Find `ArchetypeExplorerProject` in the list. You can view, edit, or delete it.
- *Note:** Clearing your browser's cache will remove this data.

10. Added value – The openEHR class glossary

10.1. Problem Statement

The openEHR specifications are complex and distributed across multiple webpages, making them difficult to navigate. For example, discovering all attributes of a class, including inherited ones, often requires switching between pages, which can quickly become confusing. Additionally, research into classes and their attributes is typically unidirectional. If a user wants to perform a reverse search—such as finding all classes that contain an element of class **PARTICIPATION**—this is essentially impossible with the current resources.

10.2. Existing Solutions

Tools like [NeoEHR UML](#) improve the visibility and navigability of the specifications by providing UML diagrams. However, these tools still have limitations, particularly regarding advanced search capabilities.

A text version of the reference model is available in the openEHR GitHub repository: [openEHR/specifications-ITS-BMM](#) but it does not contain all the information required for comprehensive search and navigation.

10.3. Proposed Solution

To address these issues, we generate a comprehensive reference model and store all relevant information in a file in a JSON compliant format. This format is easy to parse and enables a variety of search operations, including reverse lookups.

10.4. Implementation

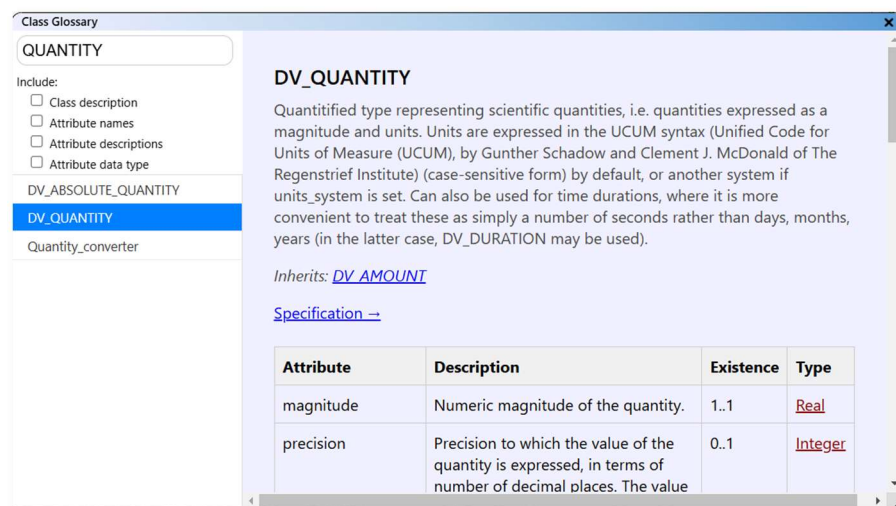
1. Data Extraction:

- A Python script connects to the following openEHR specification webpages:
 - 'https://specifications.openehr.org/releases/BASE/latest/resource.html'
 - 'https://specifications.openehr.org/releases/BASE/latest/foundation_types.html'
 - 'https://specifications.openehr.org/releases/BASE/latest/base_types.html',
 - 'https://specifications.openehr.org/releases/RM/latest/common.html',
 - 'https://specifications.openehr.org/releases/RM/latest/data_structures.html',
 - 'https://specifications.openehr.org/releases/RM/latest/data_types.html',
 - 'https://specifications.openehr.org/releases/RM/latest/demographic.html',
 - 'https://specifications.openehr.org/releases/RM/latest/ehr_extract.html',
 - 'https://specifications.openehr.org/releases/RM/latest/ehr.html',
 - 'https://specifications.openehr.org/releases/RM/latest/integration.html',
 - 'https://specifications.openehr.org/releases/RM/latest/support.html']
- The script extracts all tables from these pages. If a table represents a class, all relevant information is parsed and stored.

- The script “**openEHR_convert_specification_HTML_to_JSON_v0_9.py**” can be found in the projects GitHub page https://github.com/martinkochdesign/archetype_companion/blob/main/openEHR_convert_specification_HTML_to_JSON_v0_9.py
- The extraction process is automated to update the glossary dataset on a monthly basis. The GitHub Action is defined in the corresponding YAML file: https://github.com/martinkochdesign/archetype_companion/blob/main/.github/workflows/update-glossary-data.yml

2. Glossary Search Interface:

- A dedicated HTML page was added to the Archetype Companion tool, enabling users to search the reference model data.
- Search capabilities include class name, class description, attribute name, attribute description, and attribute data type.



The Class Glossary search interface

3. Access to the Class Glossary

- The class glossary can be accessed via a dedicated button in the upper right corner of the main view.
- Apart from the direct access, the class glossary can be opened by clicking on the data type in either the archetype information view or the template planner view.

10.5. Features

The solution offers a comprehensive and up-to-date dataset of the openEHR reference model, ensuring users always have access to the latest information.

Monthly updates are automated through Python scripting, minimizing manual effort and keeping the data current. Advanced search functionality is included, allowing for both standard and reverse lookups to enhance usability.

Additionally, the system integrates seamlessly with the Archetype Companion, providing users with easy and efficient access to the dataset.

10.6. Detailed description of use

Please refer to the Glossary section of the manual section.

11. First user test (Soft launch)

11.1. Summary of Volunteer Responses: Archetype Companion User Testing

Methodology

For the evaluation of the Archetype Companion, I conducted a structured user testing programme targeting openEHR professionals. The methodology for recruitment and participation was as follows:

1. Target group

Participants were recruited from:

- Professionals who work with and model openEHR archetypes.
- Individuals with a small project they wished to plan using the Archetype Companion.
- Volunteers willing to allow their completed project (in anonymous form) to be used as an example for future users (optional, not mandatory).

2. Recruitment and application process

- Invitation was published on **10 November 2025** on the openEHR Discourse forum.
- Prospective testers were first asked to watch a short introduction video (under 7 minutes) to understand the purpose and basic functionality of the Archetype Companion.
- Interested candidates were invited to apply via an online application form.
- **Application deadline: 21 November.**

3. Selection and onboarding

- From the applicants, a group of testers was selected to participate in the study.
- Selected participants received detailed instructions and access to the application.
- **Acceptance notifications sent: Wednesday, 26 November.**

4. Testing period

- Participants used the Archetype Companion to plan their own small openEHR-related project.
- The testing was planned to be conducted over a defined **3-week period from 1 to 19 December.**

5. Data collection and feedback

- At the end of the test period, participants were asked to complete a short feedback questionnaire.
- **Questionnaire deadline: 19 December.**
- Some participants optionally consented to have their completed project used (in anonymised form) as example material for future users.

This process provided a defined cohort of relevant professionals, a clear timeline from recruitment to feedback, and structured data collection to inform the evaluation of the Archetype Companion.

Cohort

The first round of user testing for the Archetype Companion has attracted a diverse group of **eight** volunteers, representing a range of professional roles including clinical modellers, data modellers, medical information specialists, advisers, and interoperability coordinators. The cohort's experience with openEHR varies from newcomers with less than two years of experience to seasoned professionals with over six years in the field.

The projects proposed for testing reflect a broad spectrum of clinical and technical interests. Several participants are focused on developing or refining templates for specific clinical scenarios, such as patient admission, rounds, discharge, and adverse reactions. Others are interested in applying the Archetype Companion to national projects, risk factor assessments, and specialized therapies like instillation therapy for bladder cancer and intravitreal injection therapy. Some volunteers are exploring foundational use cases in areas like imaging, medication, transfusions, critical care, and alerts or notifications.

Overall, the cohort brings together both early career and experienced professionals, ensuring a wide range of perspectives and use cases. Their collective input will provide valuable feedback on the Archetype Companion's usability and applicability across different clinical domains and project stages.

Professional Position	Years of Experience with openEHR	Project to Test with Archetype Companion
Clinical Modeller	2-4 years	Modelling the three main types of situations in which a medical professional cares for an admitted patient: admission, rounds, and discharge.
Senior Adviser	6+ years	Currently working on a project concerning instillation therapy for non-muscle invasive bladder cancer, but the project may have come too far for it to be a good testing candidate. Also starting a new project concerning intravitreal injection therapy, which may be a better candidate since it is starting from scratch.
Data Modeller	2-4 years	Works in different fields that might be interesting for this project: transfusions, critical care, or alerts and notifications.
Medical Information Specialist	0-2 years	Basic medical use cases, likely imaging or medication. Very interested in trying the Archetype Companion.
Clinical Data Modeler	0-2 years	Planning to make a template for tracking or assessment of various risk factors.
Interoperability Coordinator / Resident	0-2 years	To try and be helpful for my fellow fellow ;) Also has a dataset that needs to be templated.
Junior Analyst / Clinical Data Modeller	0-2 years	Two national projects.

Professional Position	Years of Experience with openEHR	Project to Test with Archetype Companion
Clinical Modeller	2-4 years	Adverse Reactions: Would like to use the Archetype Companion to experiment with and test potential improvements to the existing adverse reactions data model. The goal is to explore alternative structures and ensure the template supports the intended clinical and interoperability requirements.

11.2. Final survey

Of the **8** invited participants, **4** responded via the questionnaire between day 9 and 30 of December 2025.

Overall Sentiment and Key Strengths

Overall Experience

Using a 4-point scale (Poor=1, Fair=2, Good=3, Very good=4), the average overall rating was between **Good** and **Very good**, with an average of 3.25.

What Users Liked

a) Search & navigation of archetypes

“The visualization of the relationship of archetypes, and the information about archetypes (such as RM information) and the search is well done and easy.”

“Very useful for browsing & searching archetypes... especially pointing out different connections/inclusions between Archetypes.”

b) Network-like visualization & relationships

“I like the searches that can be made using the tool, using different combination and watching the relationships.”

“Visualization surprisingly responsive and appealing... Navigating the archetype dependencies in the graph was very easy.”

c) Archetype information & RM context

“I like how the information of archetypes is shown in the layout and easier to understand, comparing with CKM, for example: more intuitive, and better explained and shown.”

“Support for checklists and a smooth way to explore specifications and the Reference Model.”

d) Overall UX for clinical modellers

“Among all the tools in the openEHR ecosystem, this is the one I’ve found to be most

thoughtfully designed for clinical modellers... With the addition of a few targeted features, it has the potential to become the most valuable and widely used tool in our workflow.”

e) New archetype placeholder / template planning concept

“To be able to keep slots for archetypes not yet created in the template design mode... super useful. It hasn’t been yet embraced in other tools.”

f) Export features

“The exporter is also very useful.”

g) Tutorials and explanations

“I think tutorials and explanations are excellent.”

Conclusion

The core concept and overall direction are strongly validated.

Users see the Archetype Companion as aligned with clinical modelling workflows and solving real problems, especially around exploration and understanding of archetypes and their relationships.

Main Pain Points and Issues

Usability & Interaction Problems

a) Template Planner - element editing friction

- Users must manually add each required element from an archetype, one by one, and type each element name.
- The *element name* field in the Template Planner does not allow text selection for deletion or replacement.

“Having to manually add every required element from an archetype, one by one, and having to manually type each element name... more time-consuming and increases the risk of inconsistencies.”

“In Template Planner, text fields are not selectable for deletion. I have to keep DELETE pressed to delete a string.”

b) Difficulty understanding UI layouts and flows

- Users find it hard to get used to the different layouts and to work with the tool in a “deeper way” when designing.
- Specifically unclear: how to add archetypes to a collection and how to view which archetypes are currently in the collection.

“There are different layouts and, even if I have been trying the tool for the last week, I am still getting familiar with them and its functionalities.”

“I found it difficult to see how to add the archetypes to my collection, or to check which

archetypes are added to my collection.”

c) Browser-specific UI issues (Firefox)

- In Firefox (145.0.2), dropdown menus appear like browser “Autofill” instead of real dropdowns.
- Item labels are truncated with “...” and often unreadable.

“Dropdown menus are not displayed well: Displayed like ‘Autofill’ suggestions instead of real dropdowns, and item labels are truncated with ‘...’, making most of them not readable.”

d) Tutorial video reliability

- Tutorial videos 6 and 9 failed to load for at least one user.

“Tutorial videos 6 and 9 were not loading for me.”

e) Graph navigation ergonomics

- History navigation is powerful but not obvious; discovered through tutorials.
- Users want a simple way to go “up one step” in the dependency graph directly from the visualization.

“I discovered in the tutorial video that I can go back via the ‘History’. However, it would be useful to have a button in the visualization for going ‘upwards’ one step instead of only being able to go downwards on the dependency tree.”

Missing or Requested Features

a) Template Planner – structure loading & editing

- Automatically assign A/E typology to archetype elements.
- Load the inherent structure of the published archetype into the Template Planner (instead of building it manually).
- Option to load original element names from the archetype by default.
- Add the ability to hide elements (set occurrences to 0..0) without deleting them.
- Support value sets / terminology bindings for ELEMENTs of type CODED_TEXT.

“It would be valuable to automatically assign the A/E typology to archetype elements, or to load the inherent structure of the published archetype.”

“Adding the ability to specify value sets or terminology bindings for ELEMENTS of type CODED TEXT would greatly improve modelling precision and consistency.”

b) Export and integration with other tools

- Ability to upload the designed template to an archetype designer / another design tool.
- Download templates in OPT or OET format, or another suitable format.

“It would be useful, in the future, to find somehow to be able to upload the template designed in this tool, to the archetype designer... Or to be able to download the template designed in another format.”

“Finally, a download option for templates in OPT or OET format would be useful (although I understand this might not be the purpose of the tool).”

c) Additional visualization modes

- Mind map view for archetypes and template designs.

“Another thing that would be helpful is to have a visualization of the archetypes and templates design in a mind map view.”

d) Archetype ID support

- Automatic generation of archetype IDs from concept name and class according to openEHR conventions.

“Archetype IDs follow a specific structure. The companion could automatically generate this ID from the ‘Concept name’ and ‘Class’ fields instead of having to write it myself.”

11.3. Prioritized Improvements and Importance

The changes abstracted from the user comments can be grouped by priority:

- **P1 – Critical** Directly affects core workflow or basic usability.
- **P2 – High** Strongly improves productivity/adoption; not strictly blocking.
- **P3 – Nice-to-have** Valuable but can be scheduled after P1/P2 items.
- **P4 – Out of scope** Feature out of scope that might be of interest when the scope is re-defined.

Priority	Improvement	Rationale / Impact
P1	Fix Template Planner text fields (selection & editing)	Users cannot select text to delete or replace; must hold DELETE. This affects every edit in the Planner and creates strong frustration in a core workflow.
P1	Fix Firefox dropdown rendering & truncation	Dropdowns render like browser autofill, labels are truncated and unreadable. Firefox users cannot reliably access many configuration options; this undermines perceived quality and basic usability.
P1	Ensure all tutorial videos load reliably (esp. 6 & 9)	Tutorials are central to learning a multi-layout UI. Broken videos reduce learnability and directly contradict the otherwise positive impression of the tutorials.
P2	Auto-load archetype structure in Template Planner	Instead of manually adding every element, the Planner should import the published archetype structure and original element names. This turns the tool from a prototype into a practical daily tool for modellers.
P2	Enhance graph navigation with “up one level” control	Adds a clear navigation affordance directly in the graph view so users can go up the dependency tree without relying only on the History list.
P2	Auto-generate archetype IDs from concept name & class	Reduces cognitive load and mistakes with openEHR ID conventions; speeds up creation and encourages consistency across artefacts.
P3	Add “hide element” (0..0)	Allowing users to hide elements (set occurrences 0..0) instead of deleting them matches how template specialisation typically works, and reduces the risk of

Priority	Improvement	Rationale / Impact
	in Template Planner	losing information.
P3	Clarify & streamline “collection” workflow	It is unclear how to add archetypes to a collection and how to view the collection. This blocks a key promise of the tool: moving from search to a curated set for template design.
P3	Add mind map view for archetypes	Provides an additional high-level visualisation useful in early design and communication, especially with stakeholders less familiar with technical dependency graphs.
P3	Additional onboarding aids for multi-layout UI	Once core usability issues are fixed, add guided tours, first-time tooltips, or a “design your first template in 5 steps” overlay to further reduce learning curve.
P4	Export templates in OPT/OET and/or other interoperable formats	Makes the Archetype Companion part of the real modelling pipeline instead of a standalone exploration tool, enabling users to continue work in their usual archetype/template designers.
P4	Support value sets / terminology bindings for CODED_TEXT	Being able to define value sets and terminology bindings is essential for serious data modelling.

12. Changes implemented before final release

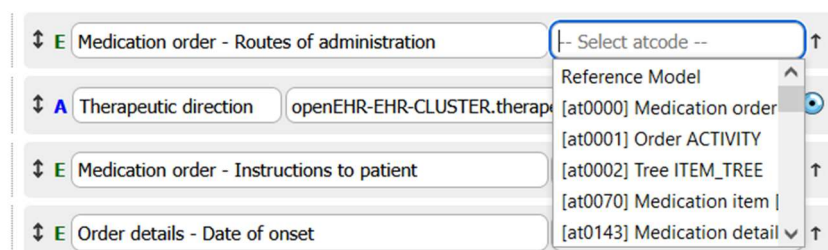
12.1. P1 - Fix Template Planner text fields (selection & editing) – SOLVED

In the template planner, the drag functionality prevented the user from selecting text in the inputs by clicking and moving the mouse. From version 0.65.1 on, the dragging is cancelled when not executed from the drag handle on the left hand side of the node.

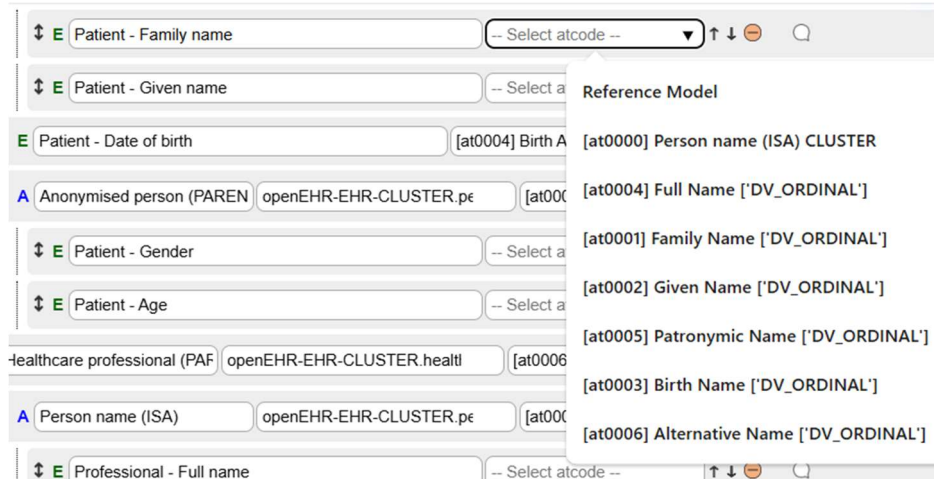
12.2. P1 - Fix Firefox dropdown rendering & truncation – SOLVED

Testing in **Firefox 1100.0.1**:

In Firefox, to open the selection, the user has to double click. Longer item names are not visible:



For comparison in Chrome, the selection box is made bigger to accommodate longer item names:



After researching it was found that Firefox does not allow to style the datalist. This is an issue that cannot be solved by CSS styling, but with a custom made handling of combi boxes in Javascript.

The handling of the dropdown combi boxes has been implemented since version 0.66.1.

12.3. P1 - Ensure all tutorial videos load reliably (esp. 6 & 9) - **SOLVED**

Tested the 9 video tutorials in URL

https://martinkochdesign.github.io/archetype_companion_test_environment/manual.html#video-tutorials

Google Chrome 132.0.6834.160:

- No problem encountered in loading all videos.

Microsoft Edge 132.0.2957.115

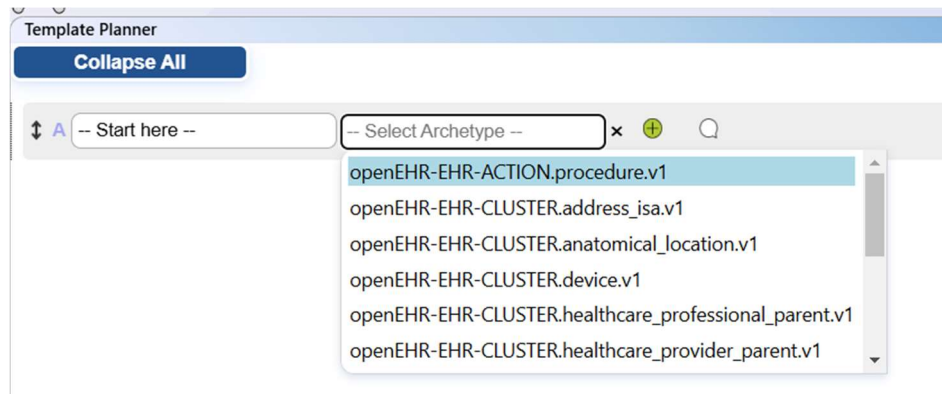
- No problem encountered in loading all videos.

Firefox 1100.0.1

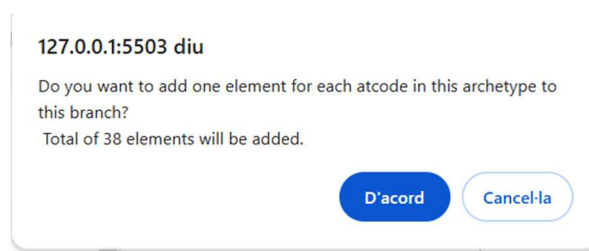
- Needs a lot of load time to load the page. This makes it seem that the videos do not load.
- To remedy the load time issue, the preload of the videos has been disabled in manual version 0.8.1 and higher.

12.4. P2 - Auto-load archetype structure in Template Planner - **SOLVED**

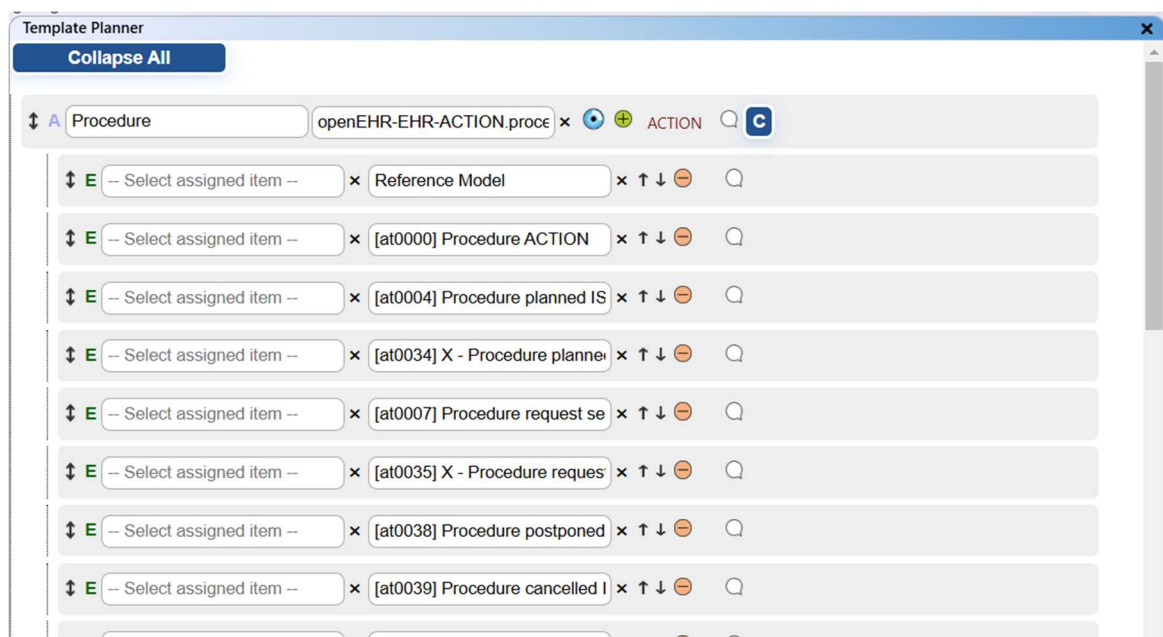
In the template planner, you can select one of the archetypes from the collection.



When doing this, a pop-up appears asking if you want to create an element under this archetype for each atcode item in the archetype.



If confirmed, the application will add an element for every item.



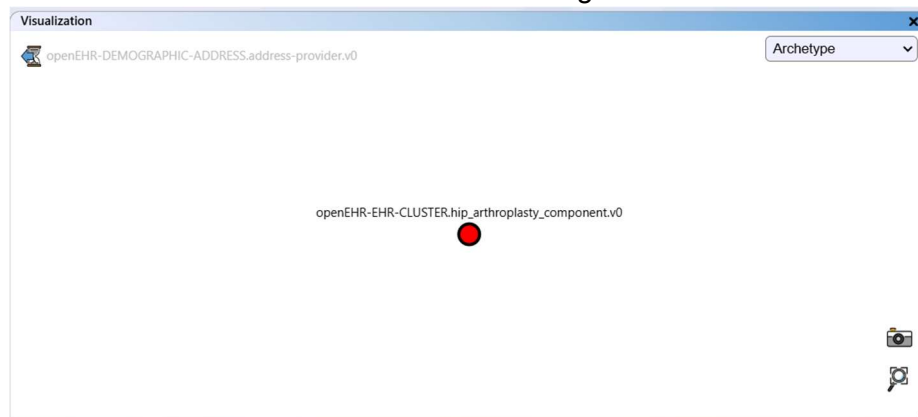
This is available since version 0.66.1.

Drawback: It has to be understood that there is no internal structure of the archetype represented here, so there is always going to be a flat list introduced here. This list might be quite long

12.5. P2 - Enhance graph navigation with “up one level” control - SOLVED

A “back” button has been added to the superior left corner of the visualization and is available

in version 0.67.0 or higher:



12.6. P2 - Auto-generate archetype IDs from concept name & class - SOLVED

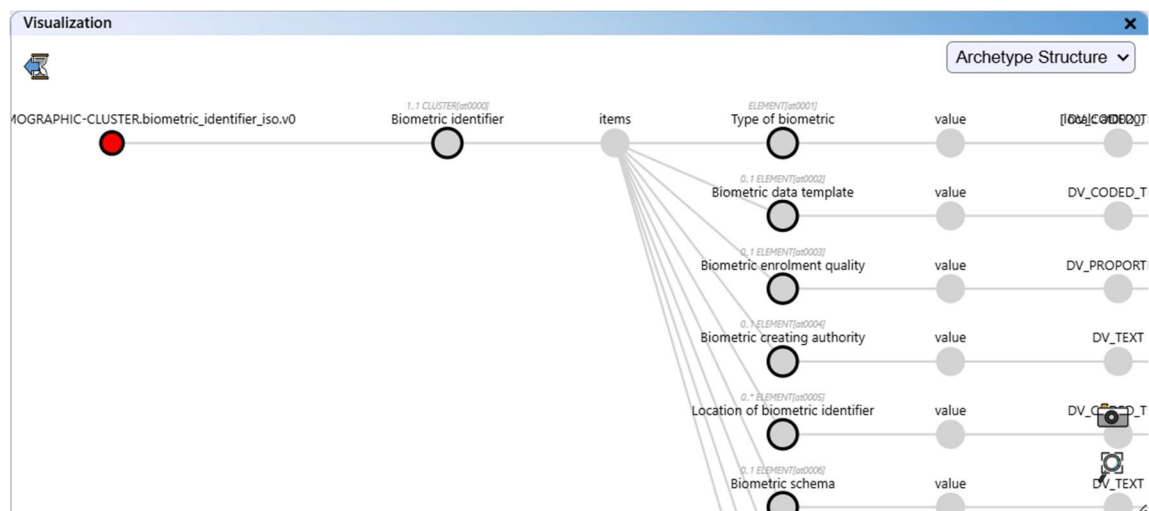
In version 0.65.0 and higher, the archetype ID name is generated automatically:

The screenshot shows a "New archetype editor" window. It contains the following fields:

- Concept name:** A text input field containing "My awesome archetype".
- Archetype ID:** A text input field containing "openEHR-EHR-EVALUATION-my_awesome_archetype_v0".
- UUID:** A text field displaying "0968626d-0611-4426-87fd-f6feb4552658".
- Class:** A dropdown menu with "EVALUATION" selected.
- Purpose:** A partially visible field at the bottom.

12.7. P3 - Add mind map view for archetypes/templates - SOLVED

Similar to the structure of the archetype that can be found in the information panel, if an archetype is selected, a new visualisation has been added to show the complete structure:



This makes it easier to see the structure of the archetype.

Additionally, the description of defining_codes have been added, so the user does not have to

look them up in the origin repository.

13. Final publication

The Archetype Companion has two different environments:

Test environment

Github URL:

https://github.com/martinkochdesign/archetype_companion_test_environment

URL of the deployed application:

https://martinkochdesign.github.io/archetype_companion_test_environment/

Production environment

Github URL:

https://github.com/martinkochdesign/archetype_companion/

URL of the deployed application:

https://martinkochdesign.github.io/archetype_companion/

14. Conclusions

14.1. Reflections on the Fellowship Programme

The openEHR Fellowship has proven to be a very valuable initiative for engaging more deeply with the openEHR community. It provided a formal framework and a clear timeframe within which to explore a focused topic, develop a tangible tool, and participate more actively in discussions around openEHR methods and practices.

Although one year initially seems like a long stretch of time, the experience has shown that it passes very quickly once the project is underway. This underlines the importance of good initial scoping and planning, and of regularly reviewing priorities throughout the Fellowship period.

Through the project, I was also able to present who I am and how I work, which opens opportunities for future collaborations. The Fellowship setting created a natural context to share my background, interests and working style, and to connect with others who may benefit from, or contribute to, this work.

Overall, my participation in the Fellowship has been clearly positive, both in terms of personal development and in terms of contributions to the openEHR community. I was particularly fortunate to have the support of my superiors, which allowed me to dedicate substantial time and attention to the project.

For a more in-depth reflection, please refer to the reflective journals that have been published during the fellowship: [Annex III – Reflective Journals](#)

14.2. Project Process and Enabling Factors

Two elements in particular made this project easier to execute and more likely to succeed.

First, I could build on previous projects that were technically similar. Having already explored related technologies and workflows meant that not everything had to be invented from scratch. This reuse of existing patterns and code lowered the technical barrier and allowed me to focus more on the domain-specific aspects of openEHR and archetype-related workflows.

Second, the project started with a relatively clear problem statement and a concrete plan. This clarity at the outset helped to keep the work focused, especially as time passed quickly over the Fellowship year. Knowing what problem I wanted to address, and having a structured idea of the steps involved, made it easier to make decisions, to prioritise, and to judge when something was “good enough” for this iteration.

Presentations and discussions during the Fellowship, especially those that provided a more profound understanding of openEHR issues and the broader context around archetype development and use, were also very helpful. They sharpened my awareness of the challenges that practitioners face and helped me position my project in relation to those challenges.

14.3. Technical Learning and Perspective

From a technical perspective, this project was a significant learning journey. I am not a

software engineer by training, which can be a limitation when creating a software-based project. Certain tasks may take longer, and best practices are not always obvious. At the same time, this non-engineering background also gave me more liberty to explore ideas creatively, to question assumptions, and to design workflows from a user-centred rather than purely technical perspective.

During the fellowship I learned a substantial amount of practical skills, including:

- JavaScript programming for implementing the core application logic
- Automation workflows in GitHub (e.g., actions and CI/CD aspects)
- Hosting webpages and making the tool accessible to others
- Using regular expressions to parse files and extract relevant information
- General experience in packaging and presenting a tool for others to test and use

These skills not only supported the current project but will also be reusable in future work, whether within the openEHR community or in other contexts.

14.4. Reflections on Community Infrastructure and Publishing

One important conclusion is that I am not fully satisfied with the current possibilities for publishing openEHR-related projects and tools to the community. While there are channels to share code, documentation and ideas, it does not yet feel easy or natural to showcase small tools, experiments, or workflow prototypes and get structured feedback from the wider community.

This raises the question of whether a more dedicated place or mechanism could be created for presenting openEHR projects - especially prototypes and experimental tools - so that they can be more visible, testable, and reusable. A more systematic way of publishing and discovering such work could encourage innovation, reduce duplication, and help good ideas spread faster.

14.5. Contribution of the Project and Future Directions

The Archetype Companion, as developed in this Fellowship, should be seen as a testing ground for ideas around the openEHR workflow rather than a final, polished product. The project contains both good ideas and less successful ones. This is natural for an exploratory effort, and it is one of its strengths: it exposes what works well in practice and what does not.

It would be valuable to continue working on this tool beyond the Fellowship period. A potential “second version” could focus on preserving and refining the good ideas that emerged, while discarding or redesigning the less effective ones. This iterative improvement could lead to a more robust and user-friendly tool that more directly supports real-world openEHR modelling and review processes.

An additional contribution of the project lies in the data analysis performed on the archetype repository. The findings from this analysis may help to support a more homogeneous quality of ADL files. Insights into current patterns, inconsistencies or gaps could be used as a basis for quality guidelines, automated checks, or improved authoring workflows. This is an area where further collaboration with the community could be especially fruitful.

14.6. Overall Closing Remarks

In summary, the Fellowship has been a valuable opportunity to immerse myself more deeply in the openEHR ecosystem, to contribute a concrete tool (the Archetype Companion), and to explore new technical and conceptual territory. The combination of a clear initial problem, supportive context, and room for experimentation has led to results that are both practically useful and conceptually informative.

There is still significant potential to refine the tool, to strengthen the infrastructure for publishing and sharing openEHR projects, and to use the insights gained - especially from the archetype repository analysis - to improve the quality and consistency of artefacts across the community. These conclusions provide a starting point for further work, collaboration, and iteration beyond the formal end of the Fellowship.

15. ANNEX I - ADL Extraction Tool – User Manual

15.1. Overview

The ADL Extraction Tool automates the extraction of key metadata from openEHR archetype files (ADL 1.4), supporting both manual and automated workflows. It can process archetypes from local collections or directly from online sources such as the Clinical Knowledge Manager (CKM). The tool supports both GUI and headless (automation-friendly) modes.

15.2. Requirements and installation

Software Requirements

For the use of the source code, the following requirements have to be met:

- Python 3.11 (or compatible version)
- Required Python libraries:
 - requests
 - datetime
 - pandas
 - customtkinter

Other libraries such as os, zipfile, re, json, tkinter, shutil, and time are part of the Python standard library.

If the executable file is used, the target machine should be running a compatible version of Windows. The executable created on a specific version of Windows might not work on significantly older versions due to differences in system libraries.

Installation

1. Download the last version of the script from the repository:
https://github.com/martinkochdesign/openEHR_fellowship_2025_project/blob/main/src/
2. Install dependencies (either from your IDE or over pip):

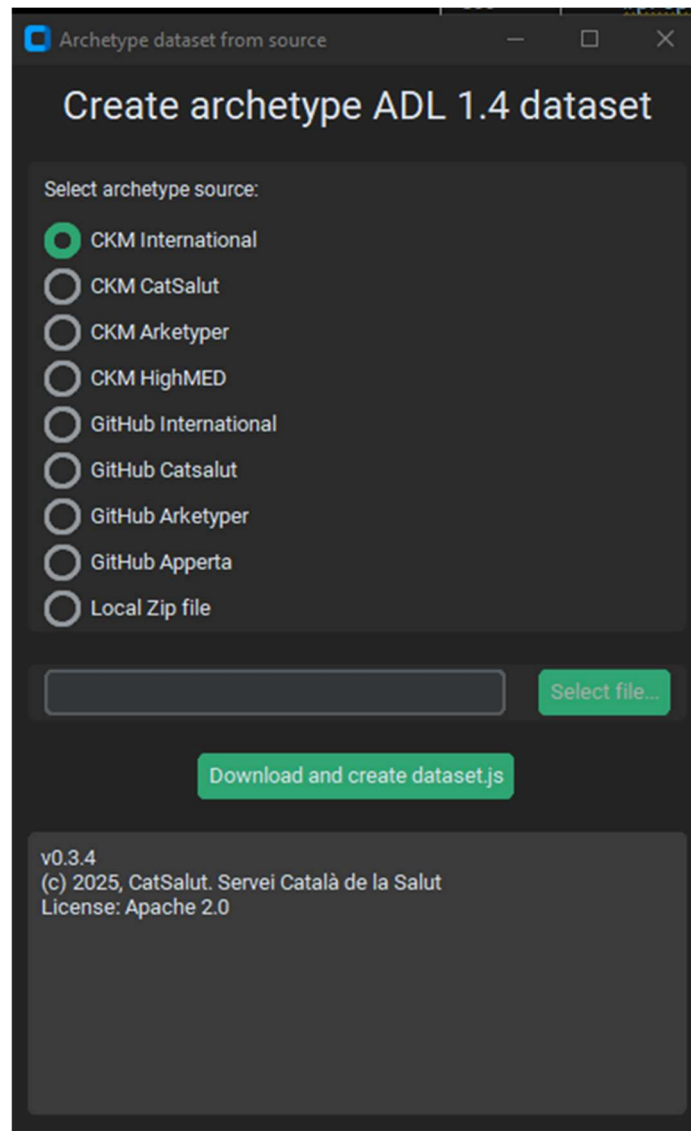
```
pip install -r requirements.txt
```

If you use the executable, you can download the latest compiled version from https://github.com/martinkochdesign/openEHR_fellowship_2025_project/tree/main/dist.

15.3. Usage

A. Running the Tool Manually (GUI Mode)

1. When launching the executable file or the script without headless mode, the GUI of the application opens.



Final GUI

2. Use the graphical interface to select the source of archetypes (local folder, CKM, or GitHub mirror).
3. Start the extraction process with the button "Download and create dataset.js". The tool will process the files and generate an output file with the extracted metadata in the same direction as the executable (or script).

B. Running in Headless Mode (Automation)

1. Set `headless = True` in the script configuration.
2. Specify the source URL (e.g., CKM REST API endpoint) and other parameters as needed:
- 3.
4. `headless = True`
5. `URL = 'https://ckm.openehr.org/ckm/retrieveResources?...'`
6. `zipFileName = 'TempZipFile.zip'`
7. `CKMorGitHub = 'CKM'`
8. `provenance = 'CKM International'`

9. Run the script.
10. The tool will run without user interaction and output the results in the same direction as the script.

15.4. Automation with GitHub Actions

The tool can be scheduled to run automatically using GitHub Actions.

Example Workflow ([.github/workflows/update-dataset.yml](#)):

```
name: Weekly Archetype Dataset Update
on:
  schedule:
    - cron: '0 3 * * 1' # Every Monday at 03:00 UTC
  workflow_dispatch:
jobs:
  update-dataset:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repository
        uses: actions/checkout@v3
      - name: Set up Python
        uses: actions/setup-python@v4
        with:
          python-version: '3.11'
      - name: Install Python dependencies
        run: |
          if [ -f requirements.txt ]; then pip install -r requirements.txt; fi
      - name: Run extractor script
        run: |
          python adl_extraction_tool.py
      - name: Commit and push updated dataset
        run: |
          git config --global user.name "github-actions"
          git config --global user.email "github-actions@github.com"
          git add dataset.js
          git commit -m "Automated weekly dataset update" || echo "No changes to commit"
          git push
```

Adjust the schedule as needed. See [crontab.guru](#) for cron syntax help.

Ensure repository Actions permissions are set to "Read and write".

15.5. Output

- The tool generates a single output file `dataset.js`, containing extracted metadata for each archetype.
- The result file is formatted for integration into a javascript code:

```
const provenance = "CKM International";
const extraction_date = "2025-07-16 07:34:54";
const allNodes = [{
  "id": "openEHR-EHR-CLUSTER.dietary_nutrients.v0",
```

```
"archetype_id": "openEHR-EHR-CLUSTER.dietary_nutrients.v0",
"class": "EHR-CLUSTER",
"lifecycle_state": "in_development",
"original_language": "en",
"date": "2016-12-14",
"translation_languages": [],
"purpose": "To record information about the nutrients consumed by an individual
for nourishment.",
"use": "Use to record information about the nutrients consumed by an individual
for nourishment. This archetype has been designed to capture measurements about
the consumption of the common groupings of nutrients. Add CLUSTER.macronutrients
and CLUSTER.micronutrients into the 'Details' SLOT in a template in order to record
consumption measurements for specific nutrients.",
"keywords": ["nutrients", "diet"],
"include": [{
  "code": "at0014",
  "label": "Details",
  "type": "CLUSTER",
  "archetypes": ["openEHR-EHR-CLUSTER.micronutrients.v0", "openEHR-
EHR-CLUSTER.macronutrients.v0"]
}],
...
```

15.6. Troubleshooting

- **Permission errors:** Ensure all files are closed before running/deleting outputs.
- **Missing dependencies:** Double-check your `requirements.txt` and Python version.
- **Automation issues:** Verify GitHub Actions permissions and workflow YAML syntax.

15.7. Support & Contributions

For issues, feature requests, or contributions, please use the [GitHub repository](#) issue tracker or submit a pull request.

16. Annex II – Archetype Designer – User Manual

Archetype Companion Manual

Author: Martin A. Koch, PhD

Copyright: 2025-2026, CatSalut. Servei Català de la Salut

Version: 0.10.0 (Last updated: 2026-02-10)

This manual and the Archetype Companion application have been developed during the

openEHR fellowship 2025 in collaboration with



16.1. Licence

This manual is licensed under the Apache Licence, Version 2.0 (the "Licence"); you may not use this manual except in compliance with the Licence. You may obtain a copy of the Licence at:

<http://www.apache.org/licenses/LICENSE-2.0>

This manual is provided "as is". The information contained herein is subject to change without notice. No warranty of any kind, either express or implied, is made in relation to the accuracy, reliability, or fitness for a particular purpose of this document. The authors and publishers shall not be liable for any errors or omissions, or for any incidental or consequential damages arising from the use of this material.

16.2. Welcome

Welcome to the openEHR Archetype Companion! This application was developed in the course of the openEHR fellowship programme 2025. This platform helps you search, plan, and visualise archetypes for your clinical modelling projects. Whether you are new or experienced, we hope that this application can help you to model more efficiently.

This tool is designed for clinical modellers and informaticians who need to find, organise, and manage openEHR archetypes. The aim is to streamline your workflow by providing a fast search for existing archetypes and visualising dependencies.

Search archetypes - You can quickly find archetypes from a database, using filters and keywords.

Visualise archetype dependencies - The visualisation features of the tool help you understand parent-child relationships, inclusions, exclusions, and similar archetypes at a glance. You can even get recommendations for similar or related archetypes.

Plan archetypes and template projects - Using the collection feature, you can organise archetypes, pre-plan new archetypes, interconnect them, and assign them to previously defined data elements of your project.

16.3. Video Tutorials

1. Overview of the application 2. GUI Tutorial 3. Project Tutorial 4. Checklist Tutorial 5. Search Tutorial Part 16. Search Syntax Tutorial 7. Search Tutorial Part 28. Archetype Placeholder Tutorial 9. Template Planner Tutorial

16.4. Example Projects

To help you get started, we provide a selection of example project files. These examples are designed for you to explore and experiment with, so you can try out features and workflows without putting your own projects or data at risk. Feel free to use, modify, or test these files as you learn more about the system.

These example projects are loosely based on existing openEHR templates. They are provided for illustrative and educational purposes only. No copyright ownership is claimed over the original openEHR templates or materials. The examples may be incomplete, simplified, or adapted, and should not be considered authoritative or comprehensive representations of the original templates. For official and complete versions, please refer to the original openEHR resources.

You can download individual example files directly:

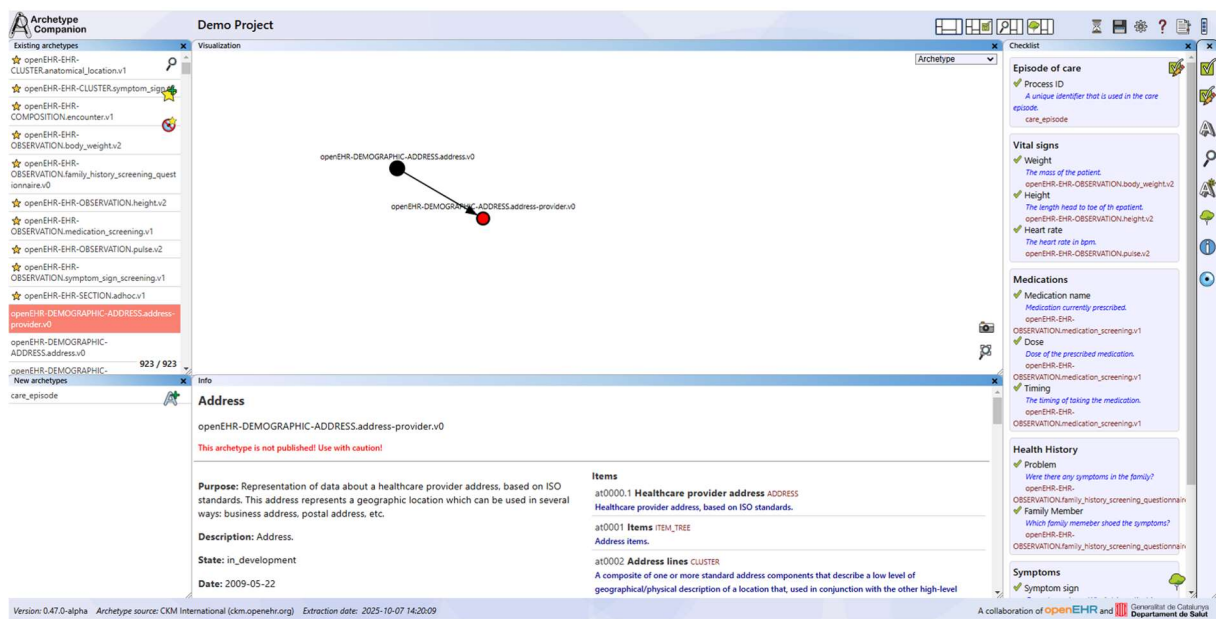
- [example_vital_signs2_project.json](#)
- [example_generic_lab_test_result_example_project.json](#)
- [example_ear_primary_hip_arthroplasty_report_project.json](#)

16.5. A brief overview of the interface

When opening the application, the user is presented with a graphical interface.

The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

Martin Andreas Koch



The archetype companion user interface.

The following image highlights the location of the current project name at the top of the interface.



The current project name in the top of the interface.

Here you can see the section of the interface that provides fast access to essential functions. These quick-access buttons allow users to perform frequent actions and access important tools without unnecessary steps.

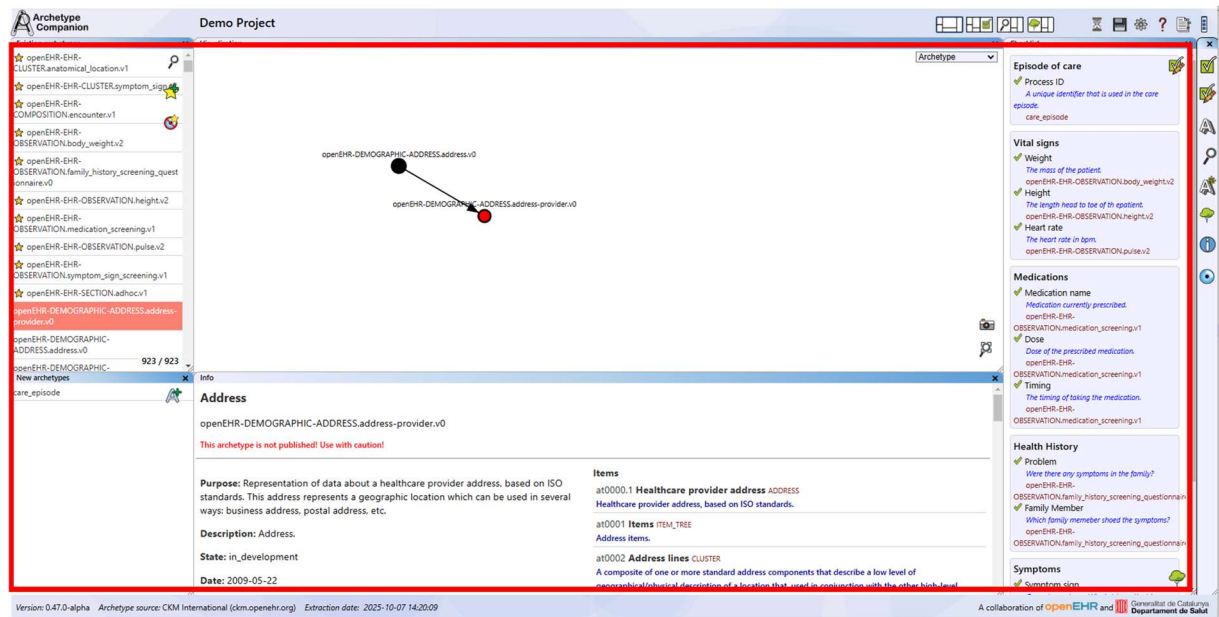


Easy access functions on the top of the interface.

In the centre of the interface you will find the workspace area, where users can organise their work using draggable and resizable windows.

The Archetype Companion - a search, visualization and planning tool for openEHR archetypes

Martin Andreas Koch



The workspace with draggable and resizable windows.

At the bottom of the interface you will encounter the version information panel. It displays the current version of the Archetype Companion tool, as well as the source and extraction date of the archetype database in use.



The version of the companion, source and extraction date of the archetype database.

Also, at the top you will find the four predefined view configurations available in the Archetype Companion. These standard views are designed to support different aspects of the application (from left to right):

- **Standard view:** The standard view, ideal for browsing and viewing information about archetypes.
- **Checklist editor view:** This view opens the checklist editor panel to the right.
- **Search view:** This view opens the search panel to the left.
- **Template planner view:** This view opens the template planner panel to the left.

Each view is tailored to a specific modelling or validation task, and users can switch between them as needed to streamline their workflow.



The 4 predefined view configurations.

For flexible use of the different windows in the interface, the various windows can be opened with the windows access bar on the right-hand side of the interface.



Access to the windows access bar.

Information about the application and the archetype data can be accessed via the "About" button in the upper right-hand corner of the main view.



169/243



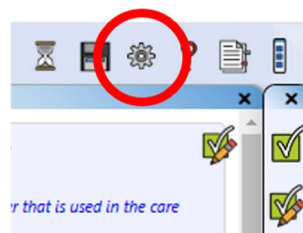
Access the manual.

The About page can be closed by clicking the "X" in the upper right-hand corner of the window.

16.7. Managing a Project

All your data in the Archetype Companion is saved within a project. Projects allow you to group archetypes and checklists of data elements for a specific modelling task.

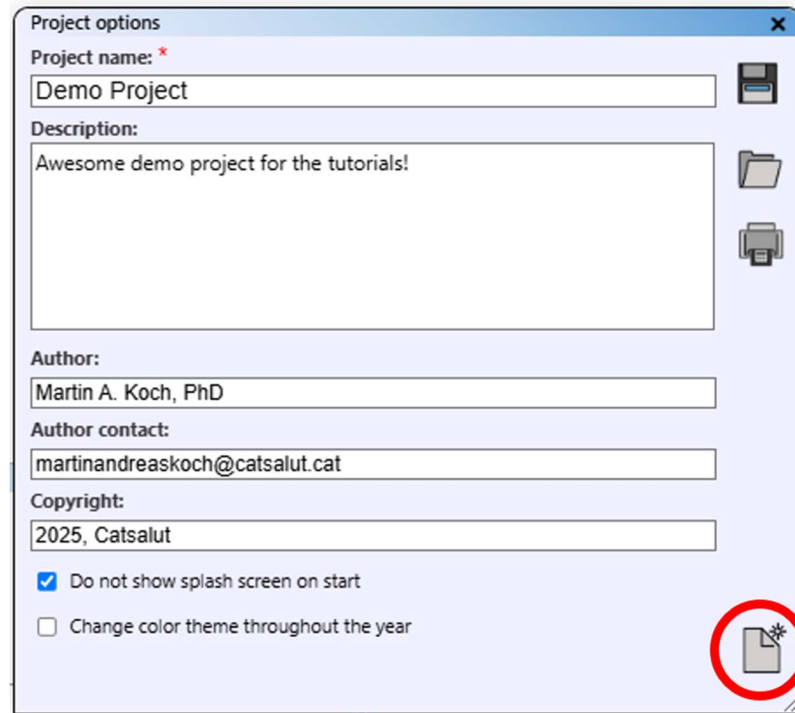
Project options are accessible from the main view of the application via the "Project options" button.



Access the project options.

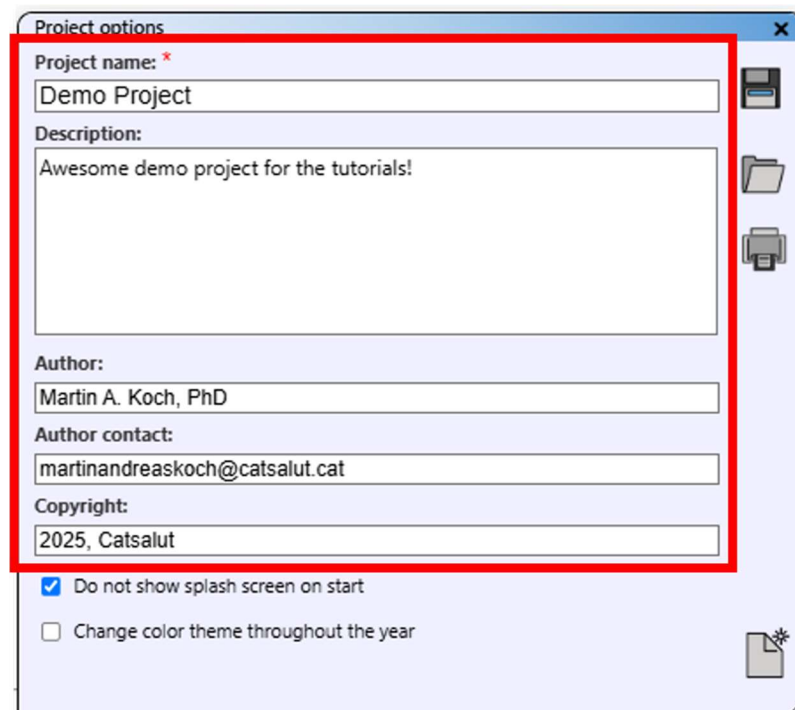
Create a Project

To start a new project, click "Create Project" on the right-hand side of the options menu. **This deletes any existing project information.**



Create a new project.

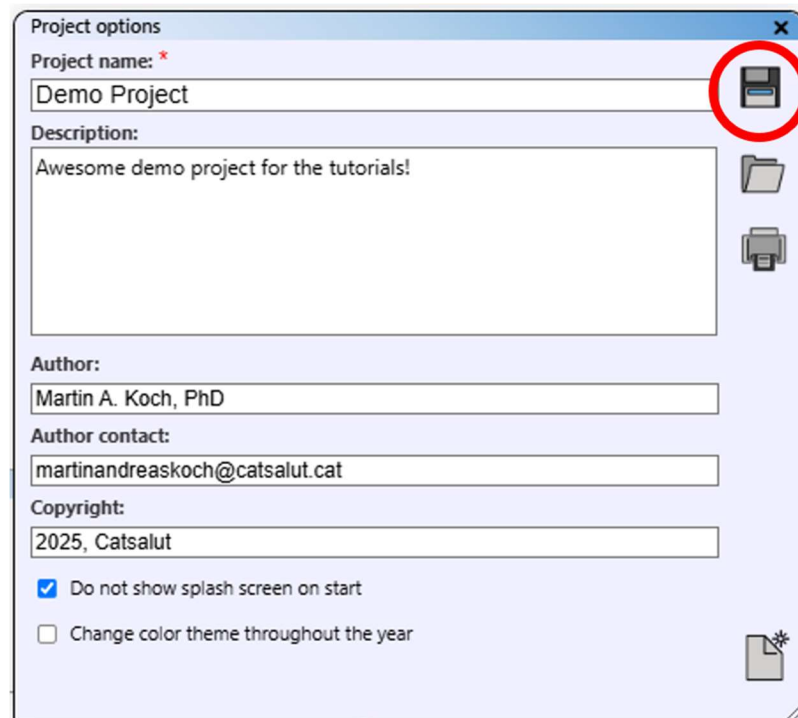
The project receives a default name, which you can change. You may also add a description, author, and copyright information.



Insert project information here.

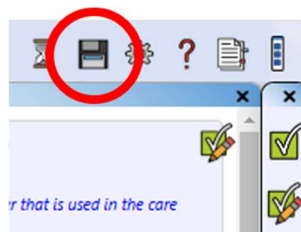
Save a Project

Your project is periodically saved to your browser's local storage. Note that this information may be lost if your browser cache is cleared. To save your project as a JSON file, use the "Save project" button in the "Project options" menu.



Save project from options window.

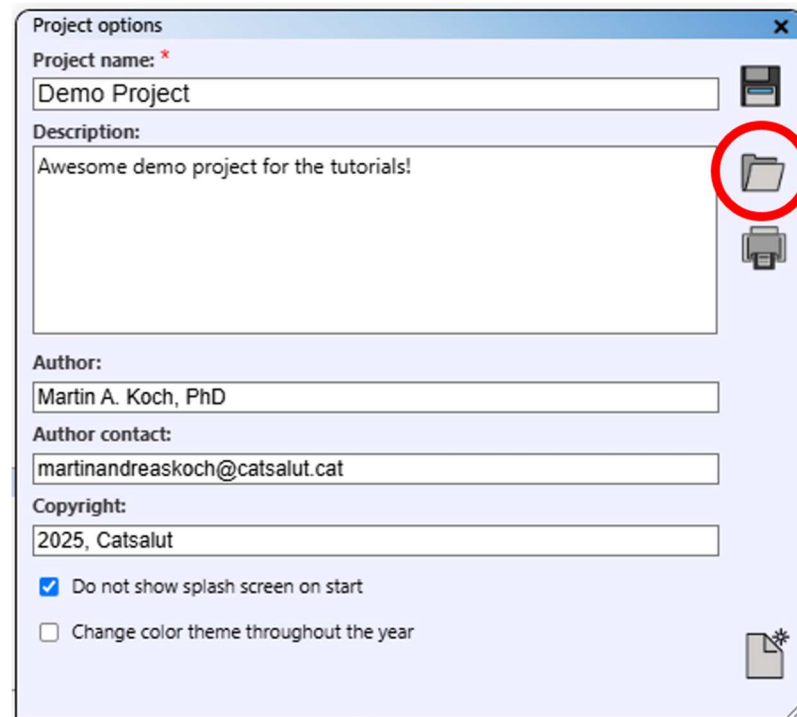
The file is saved to your browser's download directory and contains your current project state, including archetypes, checklists, and collections. The filename includes the date, time, and project name (with spaces replaced by underscores). For convenience, you can also use the "Save" button in the main view.



Save project from main view.

Load a Project

To load a saved project JSON file, open the "Project options" view and use the "Load Project from JSON File" button. Select your file in the standard file dialogue. The project will then be loaded into the application.



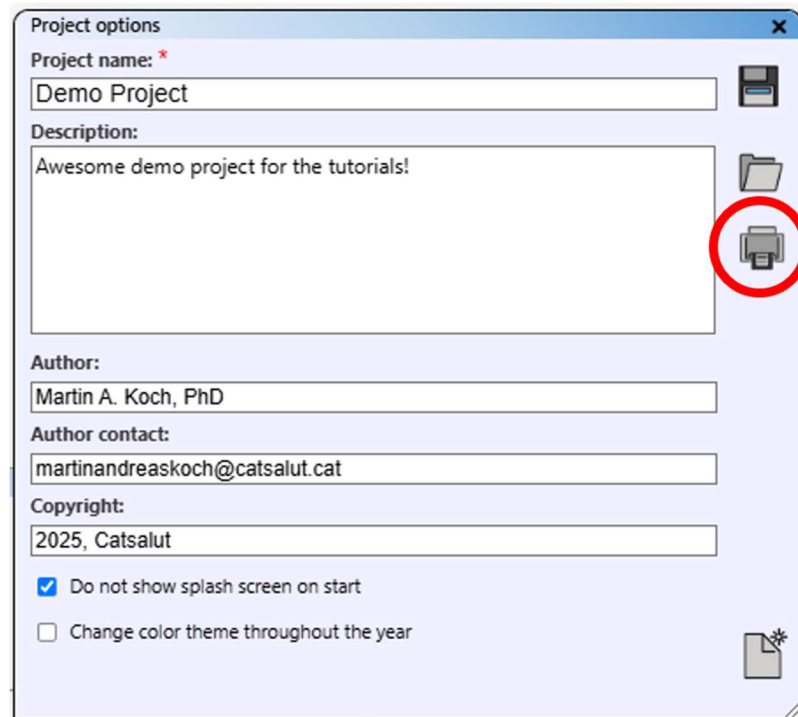
Open project from file.

Delete a Project

To delete existing project data, click "Delete/Create Project" on the right-hand side of the options menu (see "Create a Project"). This deletes any existing project information and creates a new project.

Create a Report

Generate a summary report of your project by opening the "Project options" view and clicking "Create report".



The screenshot shows a 'Project options' window with the following details:

- Project name:** Demo Project
- Description:** Awesome demo project for the tutorials!
- Author:** Martin A. Koch, PhD
- Author contact:** martinandreaskoch@catsalut.cat
- Copyright:** 2025, Catsalut
- ☒ Do not show splash screen on start
- ☐ Change color theme throughout the year

A red circle highlights a printer icon on the right side of the dialog, indicating the option to print the project report.

Create a project report.

This produces a simplified HTML view of all project data elements and assigned archetypes, which you can print as a PDF for documentation.

Archetype Companion Report

Project: Demo Project

Generated: 15/10/2025 15:10:44
Description: Awesome demo project for the tutorials!
Author: Martin A. Koch, PhD
Contact: martinandreaskoch@catsalut.cat
Copyright: 2025, Catsalut
Archetype Companion Version: 0.47.0-alpha
Database Source: CKM International (ckm.openehr.org)
Database Last Update: 2025-10-07 14:20:09

Data Element Checklist

Episode of care

Element name	Approved	Review necessary	Archetype ID
Process ID <i>A unique identifier that is used in the care episode.</i>	✓		care_episode

Vital signs

Element name	Approved	Review necessary	Archetype ID
Weight <i>The mass of the patient.</i>	✓		openEHR-EHR-OBSERVATION.body_weight.v2
Height <i>The length head to toe of the patient.</i>	✓		openEHR-EHR-OBSERVATION.height.v2
Heart rate <i>The heart rate in bpm.</i>	✓		openEHR-EHR-OBSERVATION.pulse.v2

Medications

Element name	Approved	Review necessary	Archetype ID
Medication name <i>Medication currently prescribed.</i>	✓		openEHR-EHR-OBSERVATION.medication_screening.v1

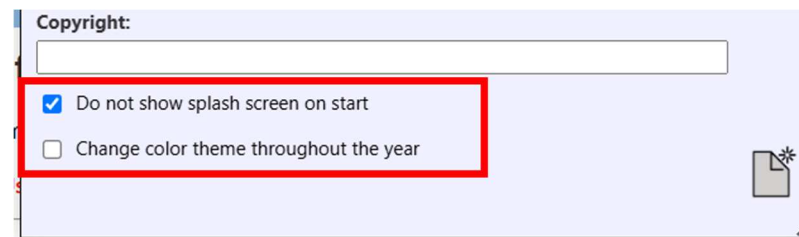
The project report.

- **Project information:** All information from the project options, including the software version and the version of the archetype data.
- **Data element checklist:** Table with all data from the checklist.
- **List of Archetypes:** List of archetypes that have been assigned to data elements from the checklist.
- **New Archetypes Details:** List of all new prototyped archetypes with all information from the editor view.
- **Planned Archetype Tree:** Visualisation of all information from the template planner in the form of a tree-like structure.

Note: When creating the report, an HTML version of the report is automatically downloaded. The filename includes the time, date, and project name (which may be modified for compatibility).

Other Project Options

Within "Project options", you can deactivate the splash screen at start-up and activate the colour theme change, which adjusts interface colours based on the time of year.



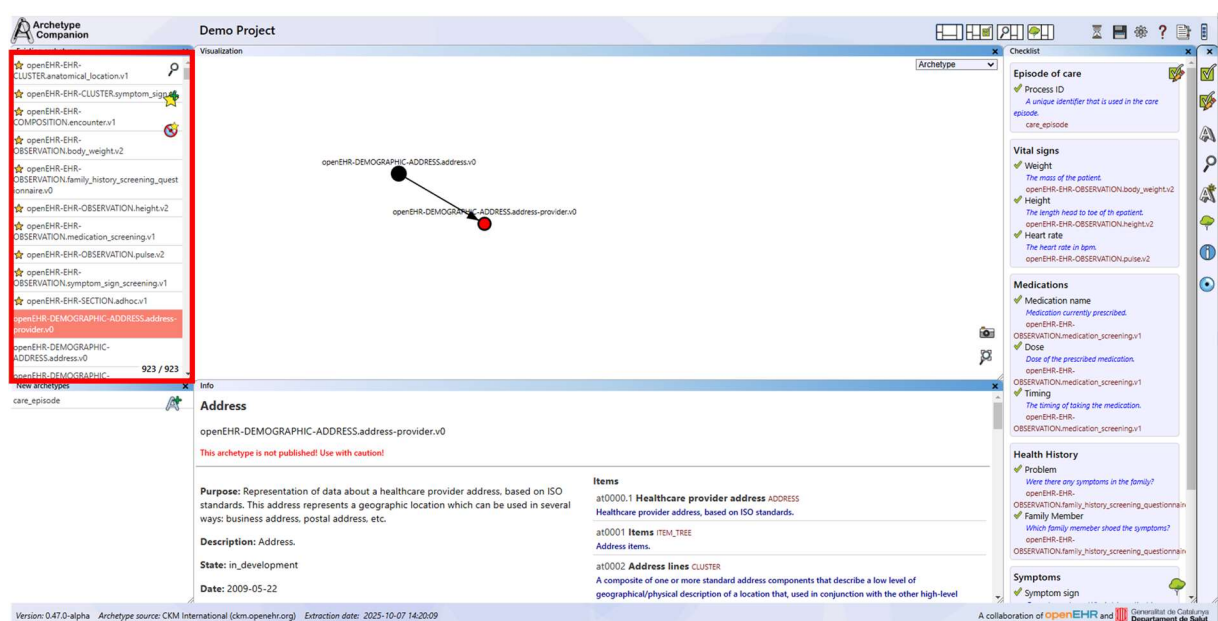
Additional options.

16.8. Navigate and Search for Archetypes

The main function of the application is browsing and searching openEHR archetypes. In addition to an easily accessible list of archetypes, this application provides a fast search engine for an openEHR archetype repository, allowing you to find archetypes relevant to your needs.

Navigating the Archetype List

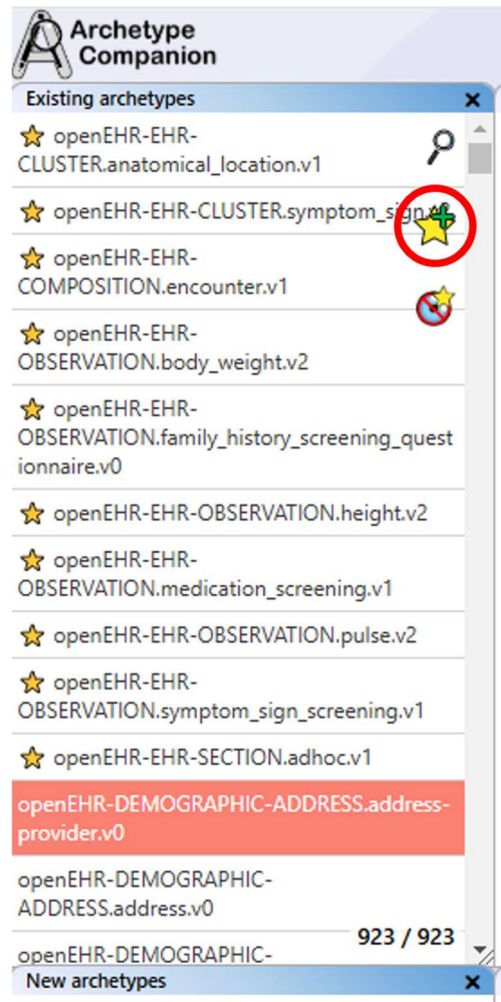
The information for archetypes extracted from a repository is presented as an alphabetised "Existing archetypes" list of archetype IDs on the left-hand side of the standard view. You can scroll through the list using the mouse wheel while hovering over it, or by using the scroll bar on the right-hand side.



Default position of the "existing archetype" list.

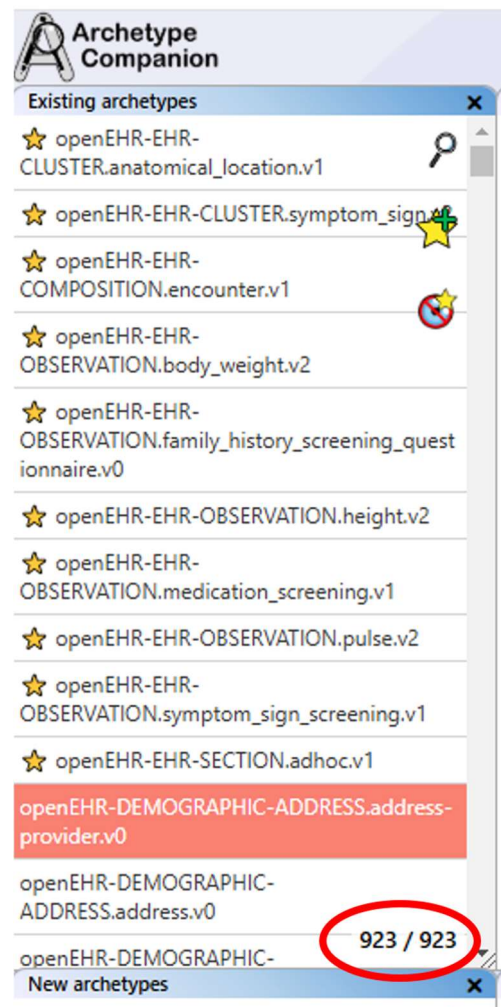
Clicking on an archetype in the list sets it as the selected archetype. As a result, the archetype's data is loaded into the information window, and the visualisation updates to the new archetype if you are in "Archetype" or "Similar" view mode. The active archetype is marked in salmon colour in the list.

The selected archetype can be added to the collection view using the "Add to collection" button. *Please refer to the Managing a Collection section for more information about collections.*



Move the selected archetype to the collection.

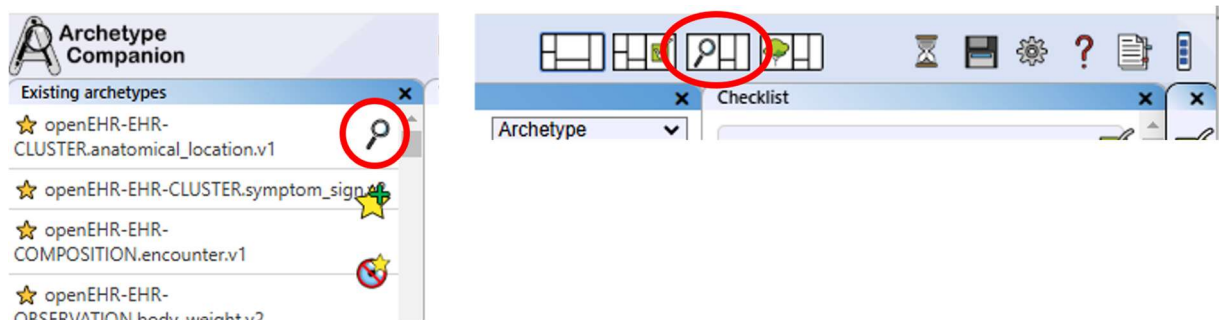
In the bottom right of the list, a number shows how many archetypes from the database are currently displayed. This number may vary depending on filters and how many archetypes have been added to the collection list.



How many archetypes from the database are in this list?

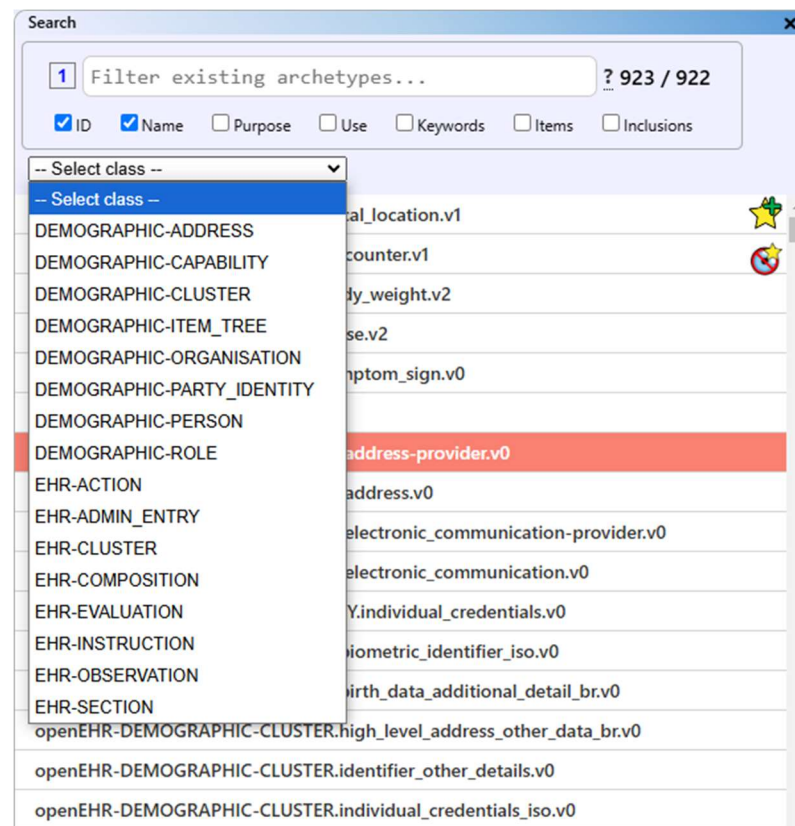
The Search View

How to Access: The dedicated archetype search can be accessed via the "Open search" button in the "Existing archetype" list or the "Search view" button at the top right of the main view. In the search view, you have the archetype list as in the "Existing archetypes" list, and you can select and transfer archetypes to the collection in the same way.



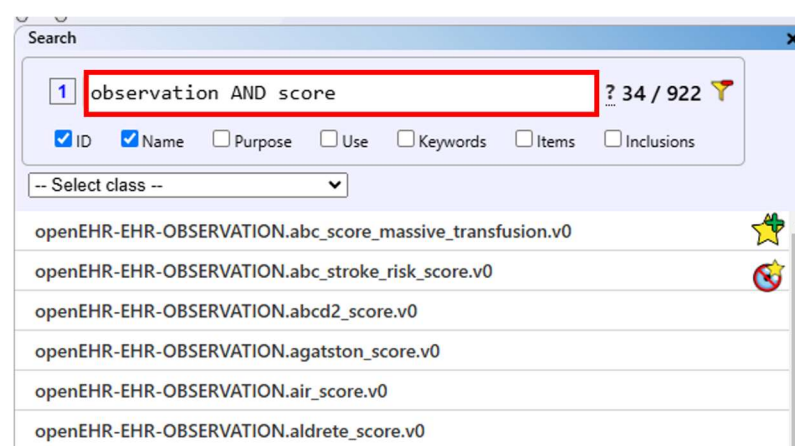
Access the search panel via the archetype list or fast access button.

Add Filter: There are different filter options. First, all archetypes can be filtered by class by selecting the desired class in the dropdown menu.



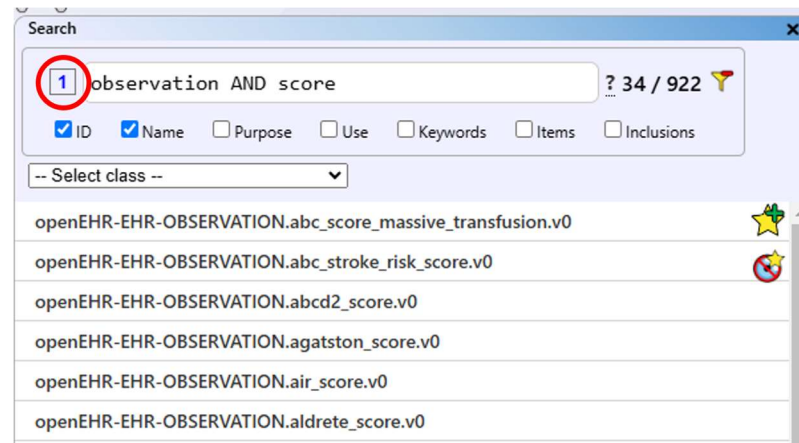
Select a class to filter all archetypes.

Second, archetypes can be filtered by text: enter your search term in the search bar, and the filter will be applied automatically with every keystroke.



Enter a search term in the search bar.

You can add multiple filters at the same time. For example, you can search for one term in the archetype name and a different term in the purpose. Select the filter by clicking on the number to the left of the searchbar.



Select another filter to create specific searches in different parts of the archetype.

Archetype Companion: Search Syntax Guide

This section explains the search syntax used in the Archetype Companion. Searching for archetypes supports basic Boolean operators, such as **AND**, **OR**, and **NOT**.

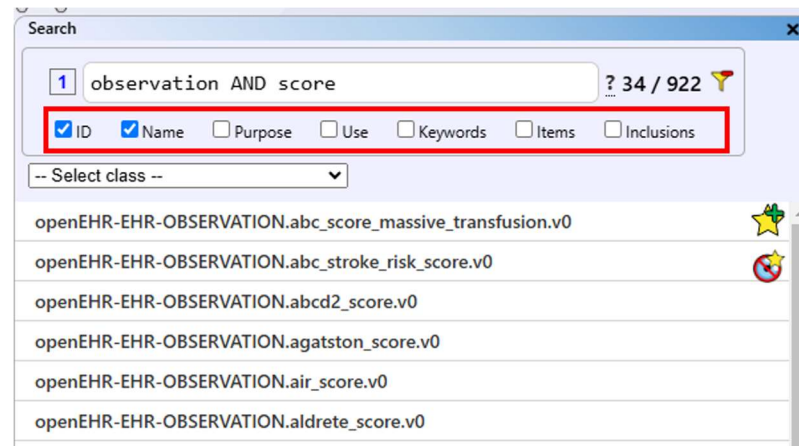
General Considerations

- **Single Term Search:** Type a term (e.g., *Location*) into the search bar. The search is not case sensitive, except for Boolean operators, which must be in uppercase.
- **Highlighting:** The searched term is highlighted in the archetype information.
- **Multiple Terms:** Add another term (e.g., *Geographic*). Both terms will be highlighted separately.
- **Phrase Search:** To search for an exact phrase, enclose the terms in quotation marks (e.g., "Geographic Location"). Only the exact phrase will be highlighted.

Using Boolean Operators

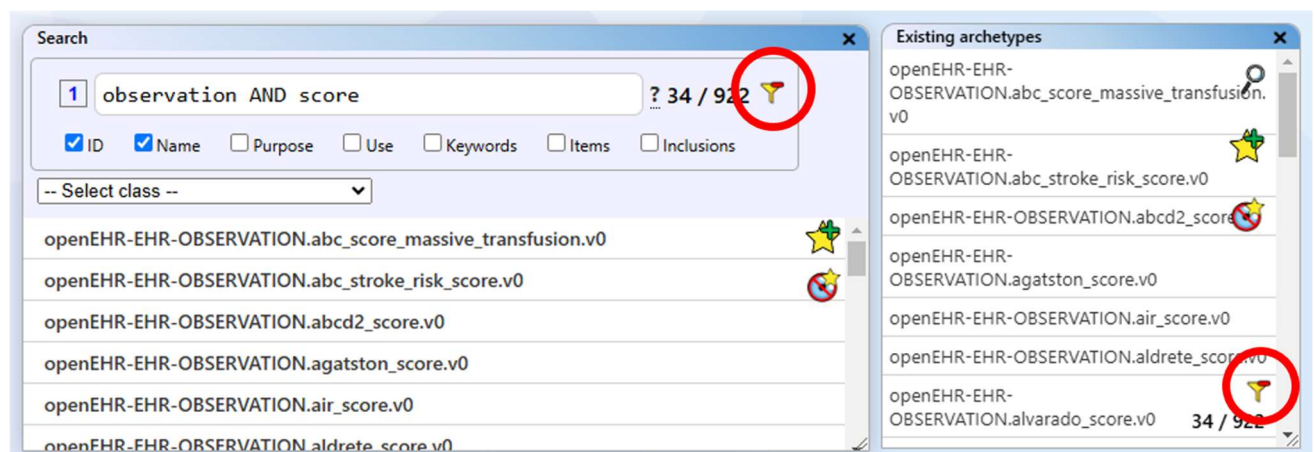
- **AND Operator:** To find archetypes containing both terms, add **AND** between them (e.g., *Geographic AND Location*). This works the same as putting a space between two terms.
- **OR Operator:** To find archetypes containing either term, use **OR** (e.g., *Geographic OR Anatomical*).
- **Combining Operators:** To combine searches, use parentheses. For example: *(Geographic OR Anatomical) AND Location*. Ensure there is a space before and after the parentheses for correct operation.
- **NOT Operator:** To exclude drafts (version zero), add **AND NOT v0** to the search bar. All version zero drafts will be excluded from the results.

You can select which attributes of the archetype to search (archetype ID, name, purpose, keywords, items and/or inclusion statements). Note that selecting all attributes may slow down the search. The number of relevant archetypes is shown beside the search bar.



Select the attributes to be included in the filter.

Delete Filter: Whenever a filter is applied to the "Existing archetypes" list, a "Delete filter" button appears beside the search bar. Clicking this button resets all filter settings. This button will also appear in the "Existing archetypes" list if a filter is active.



Resetting the filter from the search panel and "Existing archetypes" list.

Guide: Search Strategies in openEHR Archetypes for Clinical Modelling (Beginner friendly)

Recommended Order of Operation

1. Define your data element and context.
2. Search for a suitable `COMPOSITION` archetype.
3. Decide if `SECTIONS` are needed for structure.
4. Find the correct `ENTRY` archetype for your data.
5. Add `CLUSTER` archetypes for extra detail if needed.
6. (Advanced) Consider RM attributes if no archetype fits.
7. Document your choices and reasoning.

1. Understand the Goal

The aim is to find the most suitable archetype for a specific clinical data element (for example, "Blood pressure" at a patient's first GP visit). This can be challenging because:

- The archetype name may differ from your search term.
- The archetype's context or class may not match your use case.
- Sometimes, you need to combine (slot) archetypes together.

2. Start with the Big Picture: The `COMPOSITION` Archetype

What is it? The `COMPOSITION` archetype is the top-level structure for storing clinical information in openEHR.

Why start here? It defines the context (e.g., "first visit to GP") and provides slots for adding other archetypes.

- Search for `COMPOSITION` archetypes in the international CKM (Clinical Knowledge Manager).
- Read the "purpose", "use", and "misuse" sections to ensure it fits your scenario.
- Example: For a GP visit, `openEHR-EHR-COMPOSITION.encounter.v1` is often suitable.

3. Add Structure: `SECTION` Archetypes (Optional)

What are they? `SECTION` archetypes help organise your data into logical headings (e.g., "History", "Examination").

When to use? Use `SECTIONS` if it helps organise your data model, not just to match a form layout.

- Not every model needs `SECTIONS`—only add them if they make data retrieval or reuse easier.
- **Tip:** `openEHR-EHR-SECTION.adhoc.v1` is a flexible, general-purpose section archetype.

The structure of your data model does not have to match the user interface or a paper form.

4. Find the Right `ENTRY` Archetype

What are `ENTRY` archetypes? These represent the main clinical content and come in several types:

- **OBSERVATION:** For measured or reported data (e.g., blood pressure).
- **EVALUATION:** For clinical opinions, diagnoses, or assessments.
- **ACTION:** For actions performed (e.g., procedures).
- **INSTRUCTION:** For future actions (e.g., medication orders).

- **ADMIN_ENTRY:** For administrative data (not clinical content).

How to choose?

- Match the `ENTRY` type to your data's context.

Archetype Class	Definition	Type of Document	Example Clinical Document/Data
COMPOSITION	The top-level container for a clinical document or event, grouping all related content for a clinical encounter or record entry.		Discharge summary, consultation note, progress note
SECTION	A logical grouping of ENTRYs within a COMPOSITION, often used to structure documents into headings or chapters.		"History of Present Illness" section, "Medications" section
CLUSTER	A reusable structure for grouping related data elements, often used as a building block within ENTRYs or other CLUSTERs.		Blood pressure measurement (systolic/diastolic), address details, device details
OBSERVATION	Records data resulting from an observation or measurement made about the patient, often with a time series.	snapshot, history, examination, measurements, test results, scores, scales	Vital signs chart, laboratory test result, blood pressure reading
EVALUATION	Represents a clinical opinion, assessment, or summary, typically not directly measured but inferred or concluded.	opinion, synthesis, summary	Diagnosis note, risk assessment, problem list entry
INSTRUCTION	Specifies what should be done for the patient, including orders, plans, or requests for future actions.	services, therapeutics	Medication order, care plan, referral request

Archetype Class	Definition	Type of Document	Example Clinical Document/Data
ACTION	Records actions that have been performed, usually as a result of an INSTRUCTION.	services, therapeutics	Medication administration record, procedure performed
ADMIN_ENTRY	Captures administrative or non-clinical information about the patient or encounter.	administrative	Admission note, discharge summary, demographic update

- Do not pick an `ENTRY` just because it contains your data element—context matters!
- If you cannot find a perfect fit, consider requesting a new archetype or discussing changes in the openEHR community.
- **Always prefer published archetypes over drafts!**

5. Expand Details: `CLUSTER` Archetypes

What are they? `CLUSTER` archetypes add detailed data elements to `ENTRY` archetypes (e.g., a “Blood pressure” `CLUSTER` inside an `OBSERVATION ENTRY`).

- Use `CLUSTERS` to add or expand on details when the `ENTRY` archetype allows it (check for available slots).
- If you could not find your data element in an `ENTRY` archetype, try to find the `CLUSTER` first and then add it in an `ENTRY` that can include the `CLUSTER`.

6. Check the Reference Model (RM) as a Last Resort

What is it? The RM is the technical foundation of openEHR. Sometimes, data can be stored in RM attributes if not covered by archetypes.

- Only use this if you cannot find a suitable archetype or `CLUSTER`.
- This requires advanced knowledge and is rarely needed for beginners.

7. Tips for Effective Searching

- **Use Synonyms:** Archetype names may differ from your search term. Try related words or broader/narrower terms.
- **Read Descriptions:** Always check the “purpose” and “use” sections of archetypes.
- **Check Examples:** Look for example usages in CKM or community forums.

- **Ask for Help:** If stuck, reach out to the openEHR Discourse or CKM forums.

8. (Optional) Learn About Querying: AQL

AQL (Archetype Query Language) lets you search and retrieve data from openEHR repositories using archetype structures. Beginners do not need to master this immediately, but knowing it exists is helpful for future data retrieval.

[openEHR AQL Specifications](#)

Extra Beginner Tips

- Do not worry if you do not know everything—searching and checking is part of the process.
- Take notes and keep track of archetypes you consider.
- Use available tools (Archetype Companion, CKM, Archetype Designer, etc.) to browse and compare archetypes.
- Remember: The structure of your data model does not have to match the user interface or a paper form.

For more information, visit the [openEHR website](#) or the [Clinical Knowledge Manager \(CKM\)](#).

Archetype information

Disclaimer: The information about openEHR archetypes presented in this tool has been extracted automatically from external sources (such as CKM) and is provided for convenience only. We do not guarantee the accuracy, completeness, or correctness of the extracted data, nor do we claim that it is up to date. This tool does not assert or imply any copyright ownership over the archetypes themselves. For authoritative information regarding authorship, copyright, and licensing, users are strongly advised to consult the original archetype source as stated within the tool.

The Archetype Information view displays all data extracted from the archetypes found in the source repository.

The screenshot shows the 'Info' window for the 'Address' archetype. The window title is 'Info'. The archetype name is 'openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0'. A red warning message states: 'This archetype is not published! Use with caution!'. The 'Purpose' is: 'Representation of data about a healthcare provider address, based on ISO standards. This address represents a geographic location which can be used in several ways: business address, postal address, etc.'. The 'Description' is: 'Address.'. The 'State' is: 'in_development'. The 'Date' is: '2009-05-22'. The 'Keywords' are: 'demographic service, healthcare provider address, geographic location'. The 'Parent' is: 'openEHR-DEMOGRAPHIC-ADDRESS.address.v0'. The 'Children' are: 'None'. The 'Use' is: 'Used in demographic services to register data about a healthcare provider address.'. The 'Misuse' is: 'This archetype can not used for email address, IP address, computer address, telephone, fax and pager. For these items use the electronic_communication archetype.'. The 'Items' section lists: 'at0000.1 Healthcare provider address ADDRESS Healthcare provider address, based on ISO standards.', '0..1 at0001 Items ITEM_TREE Address items.', '0..* at0002 Address lines CLUSTER A composite of one or more standard address components that describe a low level of geographical/physical description of a location that, used in conjunction with the other high-level address components i.e. 'suburb / town / locality name', 'postal code', 'state / territory / province', and 'country', forms a complete geographical/physical address.', '0..1 at0021 Building/complex sub-unit type—abbreviation [DV_ORDINAL] The specification of the type of a separately identifiable portion within a building/complex, marina etc. to clearly distinguish it from another.', '0..1 at0022 Building/complex sub-unit number [DV_ORDINAL] The specification of the number of identifier of a building/complex, marina etc. to clearly distinguish it from another.', and '0..1 at0023 Address site name [DV_ORDINAL] The full name used to identify the physical building or property as part of its'.

The archetype information view.

This includes:

- Archetype name
- Archetype ID
- A warning if the archetype is not published
- Purpose
- Description
- State
- Date of authorship
- Keywords
- Parent archetype
- Child archetypes
- Use
- Misuse
- Data elements by at-code
- Inclusion statements
- Exclusion statements
- Recommended archetypes

As a special feature, you can view a collapsible representation of the archetype structure. Use the "Expand All" or "Collapse All" buttons to show or hide the entire structure at once. To expand or collapse individual elements, simply click on the desired element.



The collapsible archetype structure view.

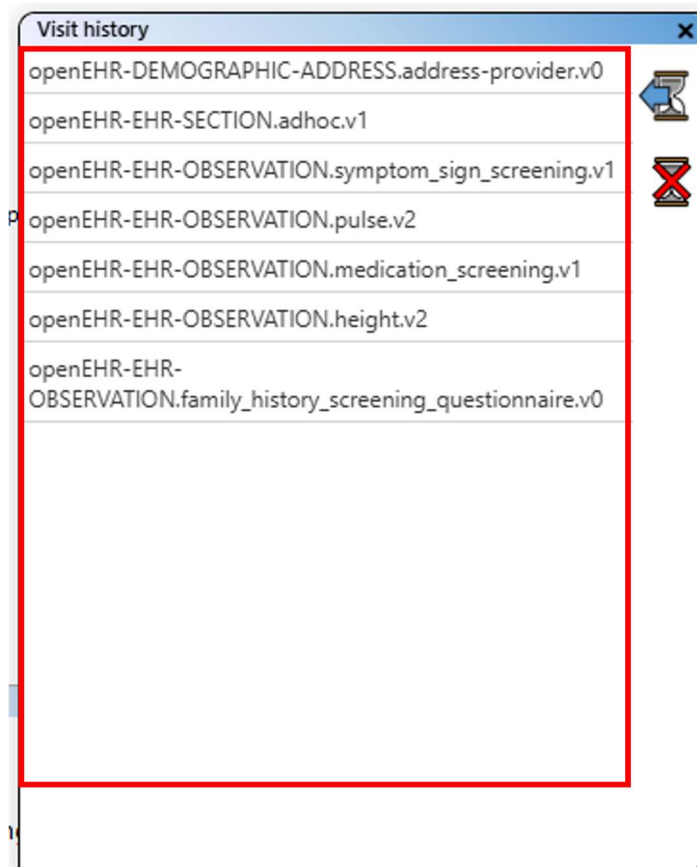
History

What is it? When navigating through archetypes, it is easy to lose track of those you have visited. The history view allows you to browse through previously visited archetypes.

How to Access: The "History" can be accessed via the "Visited archetype history" button in the upper right-hand corner of the standard view.

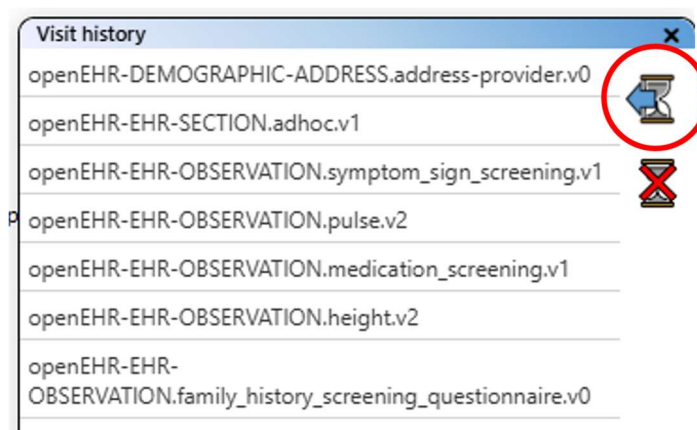


Browse History: In the "History" window, there is a list of all visited archetypes. You can scroll through the list and select an archetype to focus on it.



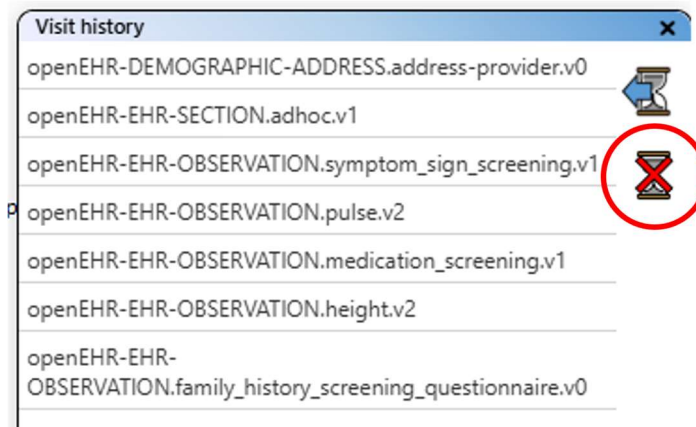
Browse the list of visited archetypes.

Go Back One Step: You can go back step by step using the "Back" button. Note that when going back, the last entry in the history is deleted.



Go back one step in the archetype history.

Delete History: You can delete all history entries by using the "Clear History" button.



Delete the list of visited archetypes.

16.9. Managing a Collection

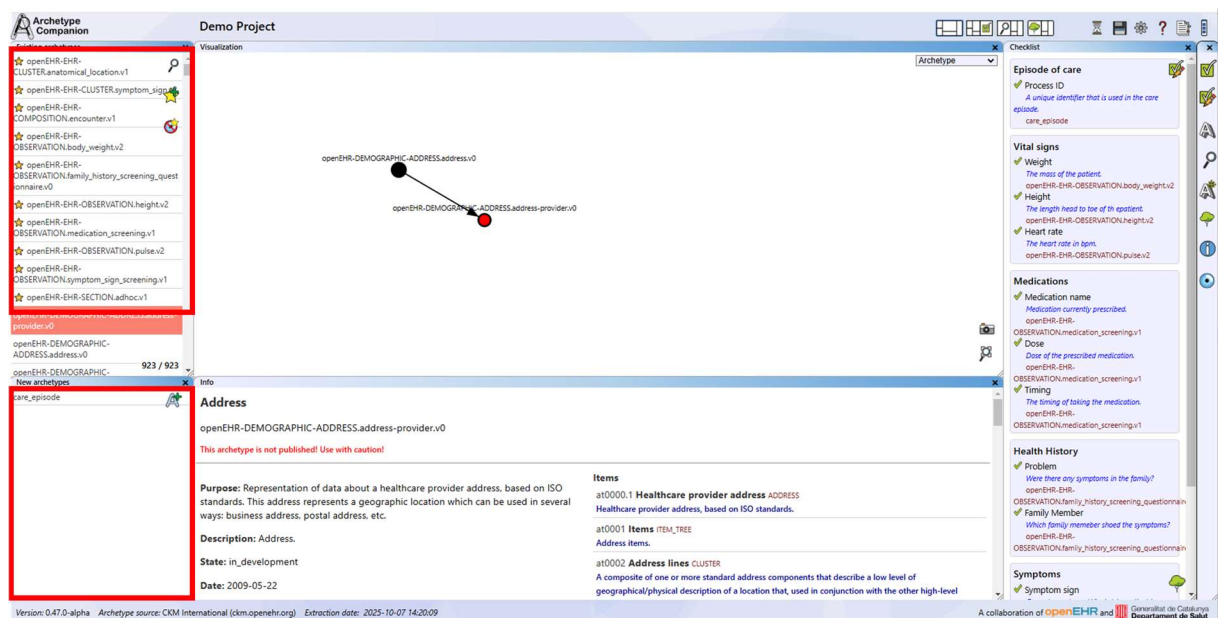
What it is – The archetype collection is a group of archetypes that you wish to keep together for reference or (re)use in your project.

A collection contains both "existing" archetypes from the archetype database and basic "new" archetypes that you can define yourself.

Where to find it – The collection is composed of existing archetypes that have been marked as collection items and all new archetype prototypes.

The existing archetypes can be found in the archetype list, marked with a star.

The new archetypes are located on the lower left-hand side of the main view interface.

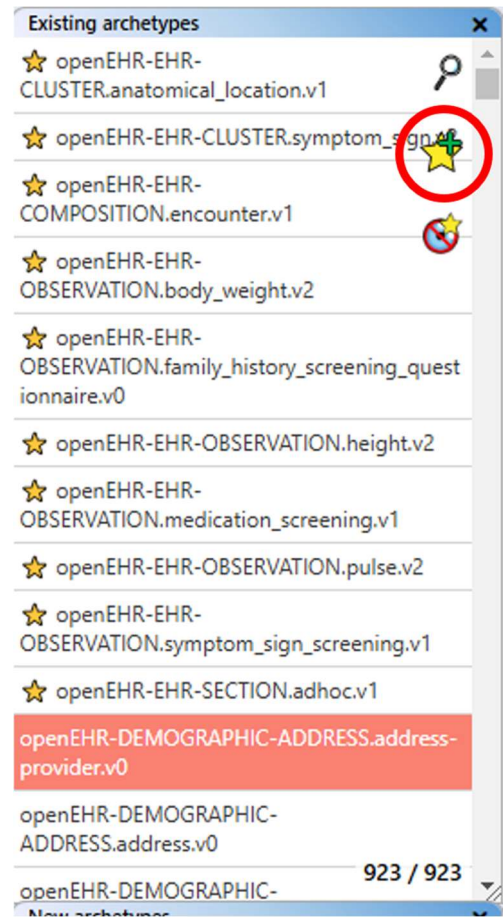


Default position of the "collection" list.

Add an Existing Archetype

Add archetypes from the "existing archetypes" list to your collection.

Select the desired archetype in the "existing archetypes" list by clicking on it. An icon will appear on the right-hand side of the list. Click the "Add to collection" icon, and the archetype will be marked as part of the "collection" list.

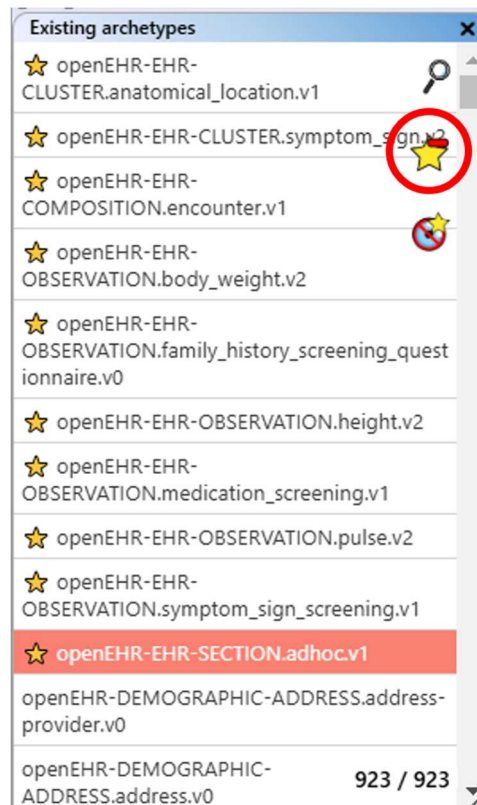


Add an "existing" archetype to the collection list.

Note: Apart from being part of the collection, the archetype is now available in the dropdown menus of the checklist editor.

Remove Existing Archetype

If you no longer wish to keep an "existing" archetype in your collection, you can move it back to the "existing archetypes" list. Click on the archetype in the collection list to select it. An icon will appear on the right of the collection list.



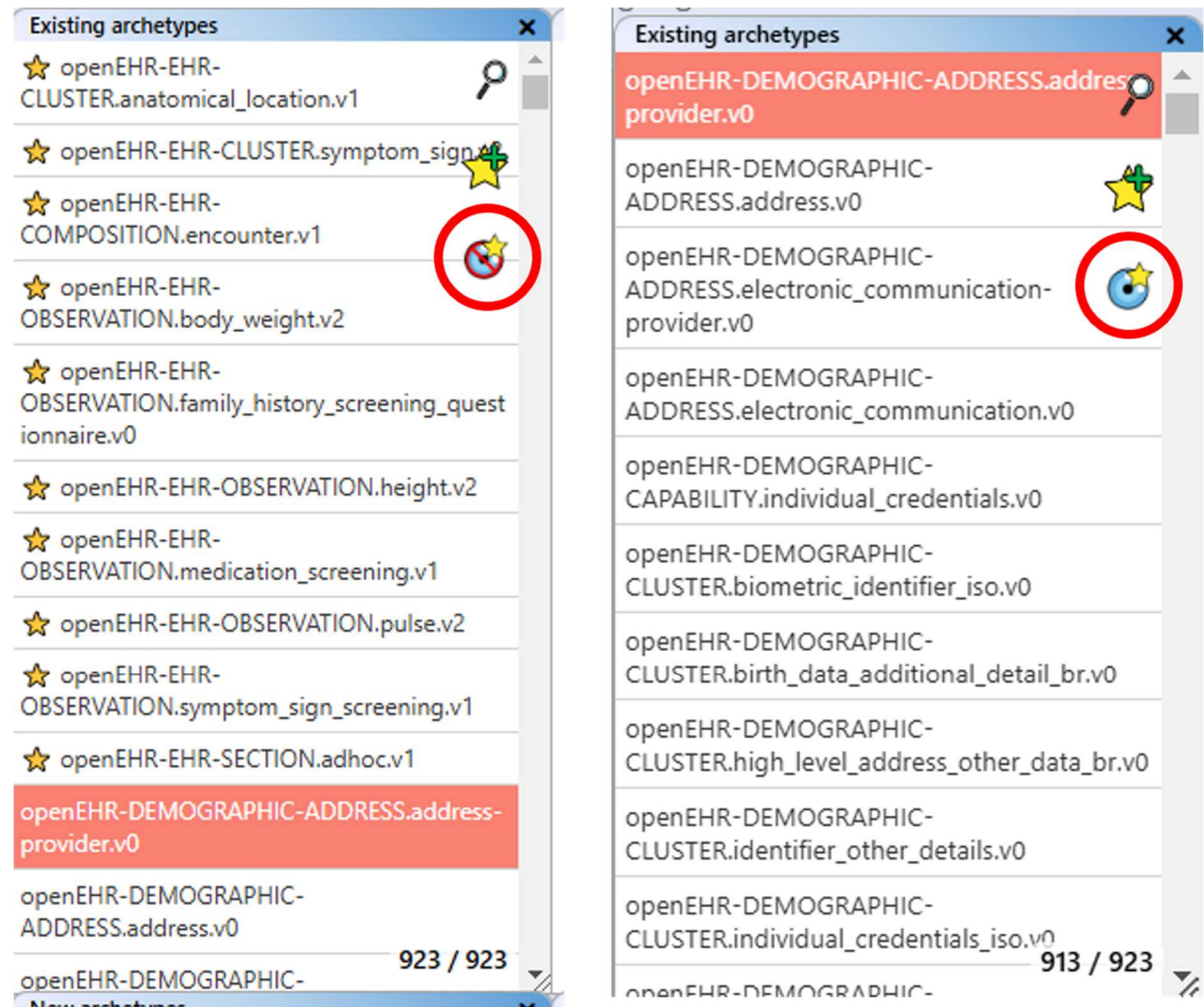
Remove an "existing" archetype from the collection list.

Click the "Remove archetype from collection" icon. The collection tag is removed from the archetype.

Hide or Show Collection Archetypes

Sometimes the accumulation of collection archetypes at the top of the existing archetype list makes the list difficult to use.

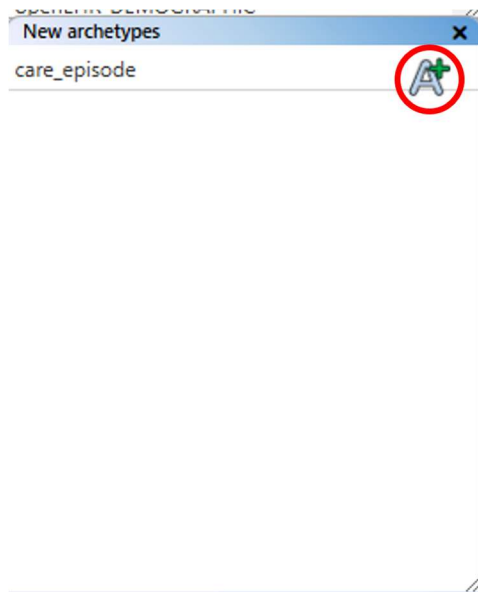
You can hide or show the collection archetypes by clicking on the hide or show icon.



Hide or show collection archetypes from the existing archetype list.

Create a "New" Archetype

If you need to plan a new archetype for your project, you can create a basic placeholder here. On the right side of the new archetype list, you will find the "Add new archetype" icon. Click on it to create a new archetype.

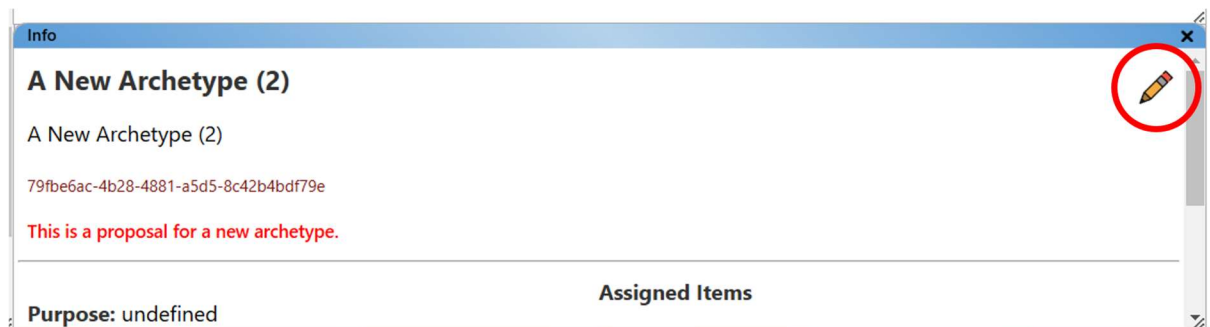


Create a "new" archetype for the collection list.

The new archetype will be selected automatically. As shown in the information view, the archetype has no information associated with it.

Edit a "New" Archetype

Select the "new" archetype you wish to edit in your new archetype list. Note that the archetype data will be loaded into the archetype info window. There is also an edit icon. Click the edit icon to open the editor window.



Edit option in the archetype info window, if a "new" archetype is selected.

Here, you can edit the concept name and the archetype ID. New archetypes are identified in the system by a unique UUID assigned during creation.

New archetype placeholder editor

Concept name:

My nice new archetype

Archetype ID:

openEHR-EHR-CLUSTER-my_nice_new_archetype_v0

UUID: f114b5eb-654e-40d0-ab2b-52a24b0f1d19

Class:

CLUSTER

Purpose:

A new archetype that will change my world!

Keywords:

awesome, nice, paradigm shift

Assigned elements:

Included archetypes:

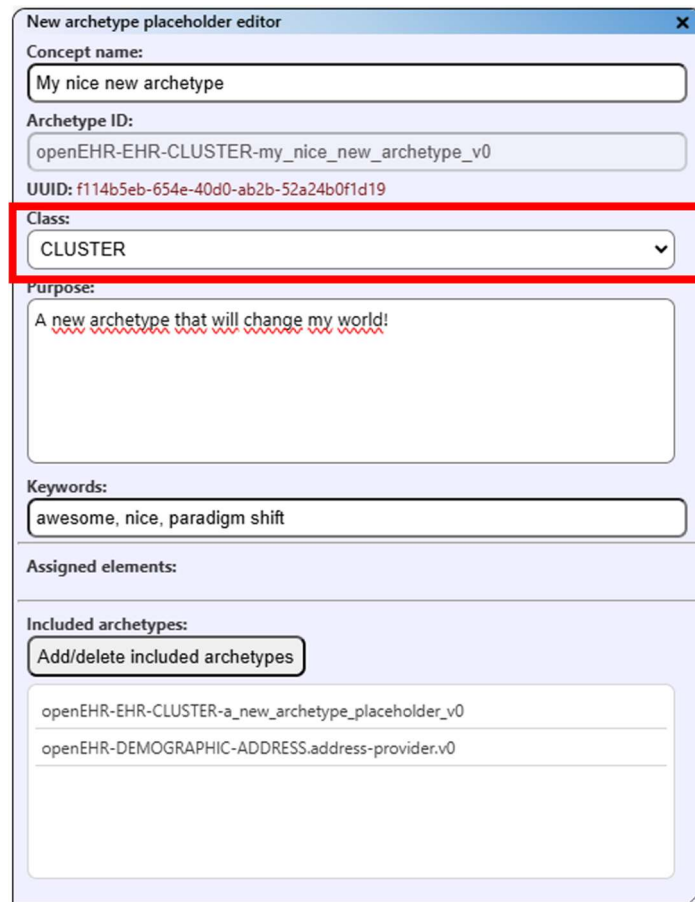
Add/delete included archetypes

openEHR-EHR-CLUSTER-a_new_archetype_placeholder_v0

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

Create an identity for the new archetype.

You may also select a class for this archetype.



The screenshot shows a web-based form titled "New archetype placeholder editor". The form contains several fields and sections:

- Concept name:** A text input field containing "My nice new archetype".
- Archetype ID:** A text input field containing "openEHR-EHR-CLUSTER-my_nice_new_archetype_v0".
- UUID:** A text input field containing "f114b5eb-654e-40d0-ab2b-52a24b0f1d19".
- Class:** A dropdown menu with "CLUSTER" selected. This section is highlighted with a red rectangular box.
- Purpose:** A text area containing the text "A new archetype that will change my world!".
- Keywords:** A text input field containing "awesome, nice, paradigm shift".
- Assigned elements:** A section header with no content below it.
- Included archetypes:** A section with a button "Add/delete included archetypes" and a list of two archetypes: "openEHR-EHR-CLUSTER-a_new_archetype_placeholder_v0" and "openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0".

Select a class for the new archetype.

To finish, you can add a purpose and keywords to the new archetype.

New archetype placeholder editor

Concept name:
My nice new archetype

Archetype ID:
openEHR-EHR-CLUSTER-my_nice_new_archetype_v0

UUID: f114b5eb-654e-40d0-ab2b-52a24b0f1d19

Class:
CLUSTER

Purpose:
A new archetype that will change my world!

Keywords:
awesome, nice, paradigm shift

Assigned elements:

Included archetypes:
Add/delete included archetypes

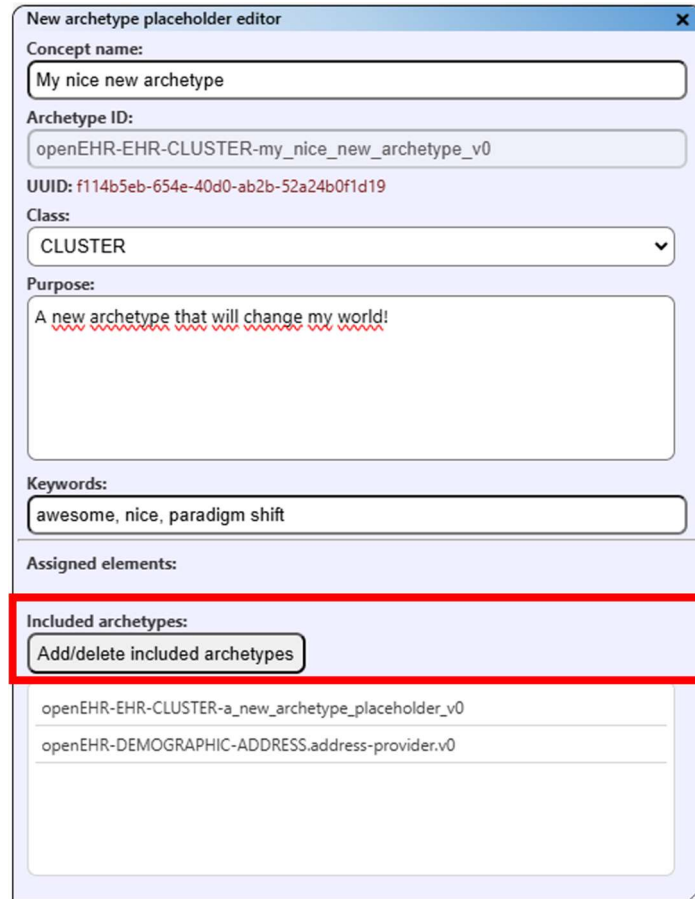
openEHR-EHR-CLUSTER-a_new_archetype_placeholder_v0
openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

Adding more detail to the "new" archetype.

Please note that this "new archetype" does not constitute a "real" openEHR archetype. It is simply a tool to indicate the need for a new archetype in the repository.

Include Archetype

If you plan to have a Cluster in the "new" archetype that is defined to include, you can add a list of inclusion statements. To do so, click on the "Add/delete included archetypes" button above the included archetypes list.



New archetype placeholder editor

Concept name:
My nice new archetype

Archetype ID:
openEHR-EHR-CLUSTER-my_nice_new_archetype_v0

UUID: f114b5eb-654e-40d0-ab2b-52a24b0f1d19

Class:
CLUSTER

Purpose:
A new archetype that will change my world!

Keywords:
awesome, nice, paradigm shift

Assigned elements:

Included archetypes:
Add/delete included archetypes

openEHR-EHR-CLUSTER-a_new_archetype_placeholder_v0

openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0

Select an the button to include (or delete) an archetype from the dropdown list.

In the opened list you can see all archetypes in your current archetype collection. Archetypes that are already in the inclusion list are marked in red, archetypes that are available for inclusion in green. To **delete an archetype** from the inclusion list, just click on its name in red. To **add an archetype** to the inclusion list, click on its name in green.

New archetype placeholder editor

Concept name:
My nice new archetype

Archetype ID:
openEHR-EHR-CLUSTER-my_nice_new_archetype_v0

UUID: f114b5eb-654e-40d0-ab2b-52a24b0f1d19

Class:
CLUSTER

Purpose:
A new archetype that will change my world!

Keywords:
awesome, nice, paradigm shift

Assigned elements:

Included archetypes:
Add/delete included archetypes

- openEHR-DEMOGRAPHIC-ADDRESS.address-provider.v0
- openEHR-DEMOGRAPHIC-ADDRESS.address.v0
- openEHR-DEMOGRAPHIC-ADDRESS.electronic_communication-provider.v0
- openEHR-DEMOGRAPHIC-CLUSTER.birth_data_additional_detail_br.v0
- openEHR-DEMOGRAPHIC-CLUSTER.identifier_other_details.v0
- openEHR-EHR-CLUSTER-a_new_archetype_placeholder_v0

Add or delete the inclusion to/from the "new" archetype by clicking on its name.

Delete a "New" Archetype

If you no longer wish to keep the "new" archetype in your collection, you can delete it from the list. Click on the archetype in the new archetype list to select it. An icon will appear on the right of the new archetype list.

New archetypes

A New Archetype

care_episode

Delete the "new" archetype from the collection list.

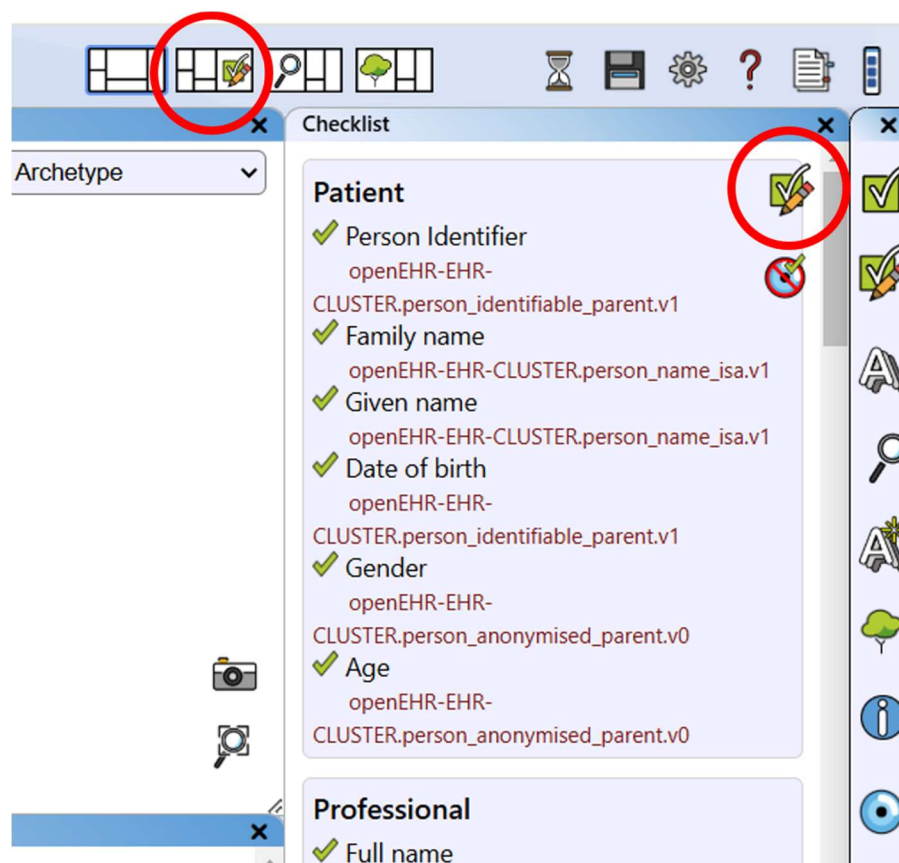
Click the "Delete new archetype" icon. The archetype will be deleted permanently and will disappear from the new archetype list.

16.10. Managing Checklists

When exploring archetypes, you may need to find specific data elements that match a list you have prepared in advance. We refer to this structured list of data elements as a "checklist". Checklists help you track required data elements and ensure nothing is missed during modelling.

Where to find the checklist

You can see the checklist on the right-hand side of the main view. When creating a new project, this view is empty. To create or edit the checklist, click the "Edit" button in the checklist view. You can also open the editor via the Checklist Editor View at the top of the interface.

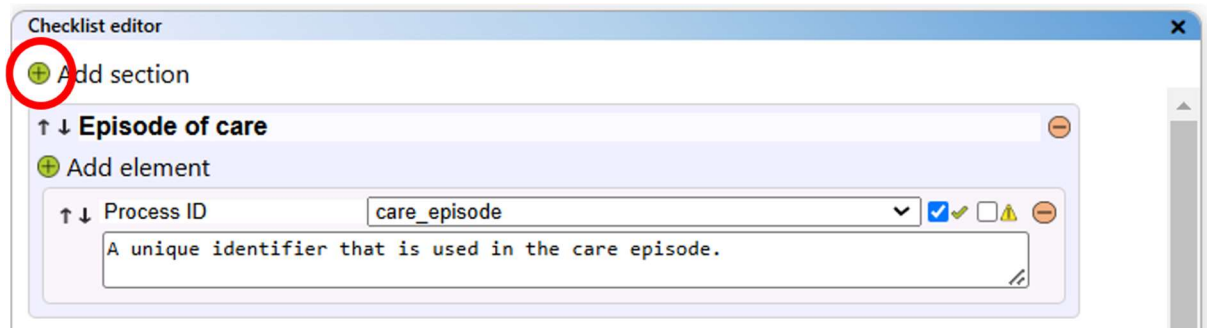


Access the checklist editor via the checklist or standard view.

The checklist is structured in sections and elements, allowing you to group data elements into sections for a basic structure.

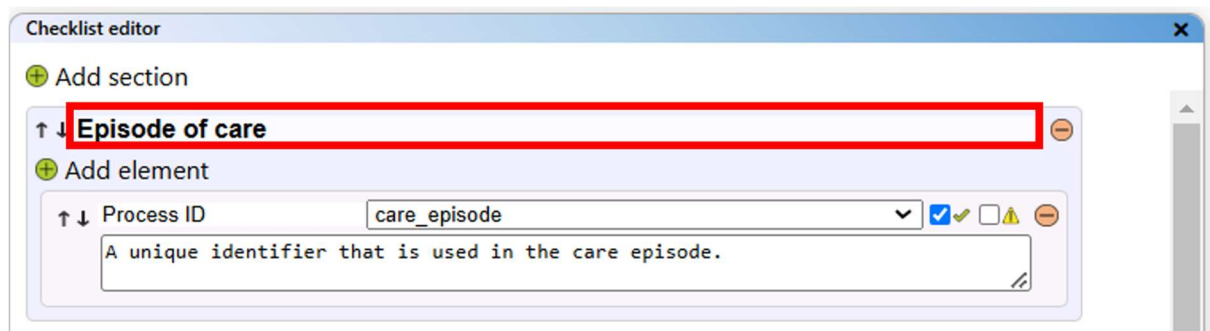
Creating a Checklist in the Editor

Add a Section: To add elements, you must first create a section by clicking the "Add Section" button.



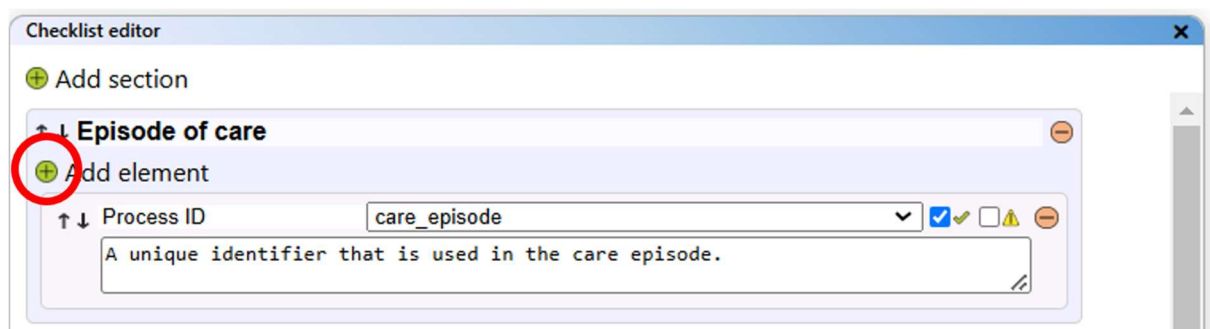
Add a section to the checklist.

You can rename the section by clicking its name and typing a new one. Please ensure each section has a unique name.



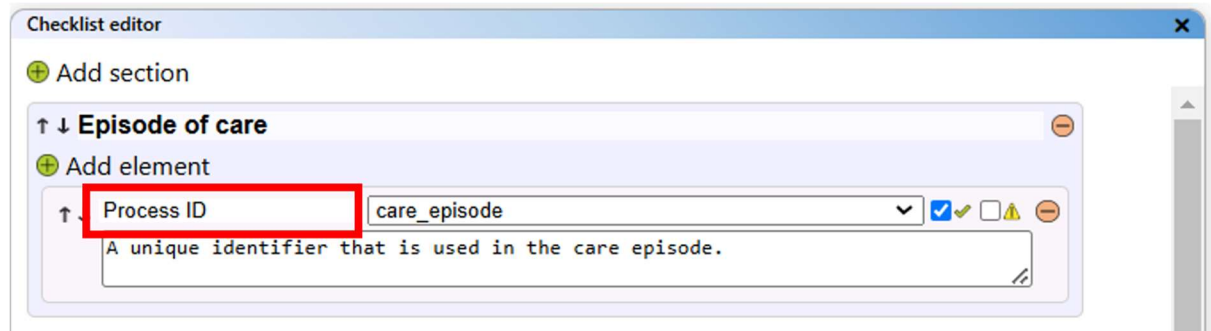
Rename the section with the corresponding input field.

Add Elements to a Section: Within a section, add individual data elements as needed by clicking the "Add element" button below the section name.



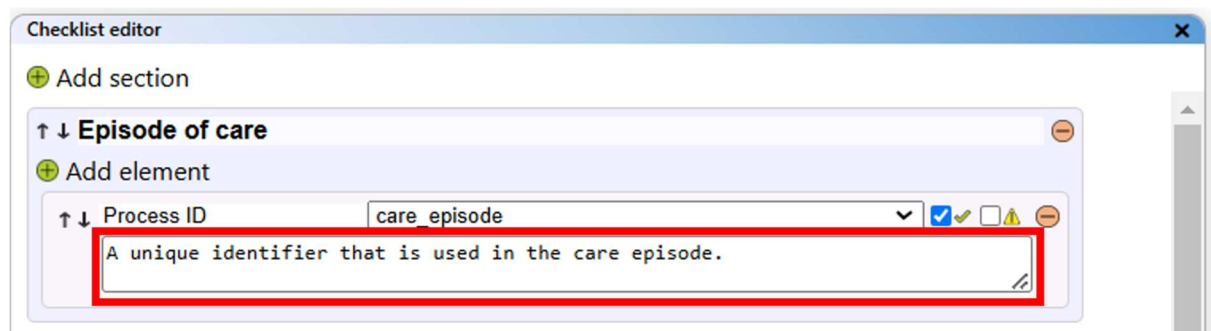
Add an element to a section of the checklist.

Each new element is automatically named, but you can change this by clicking on the name and typing a new one. Ensure element names are unique within each section.



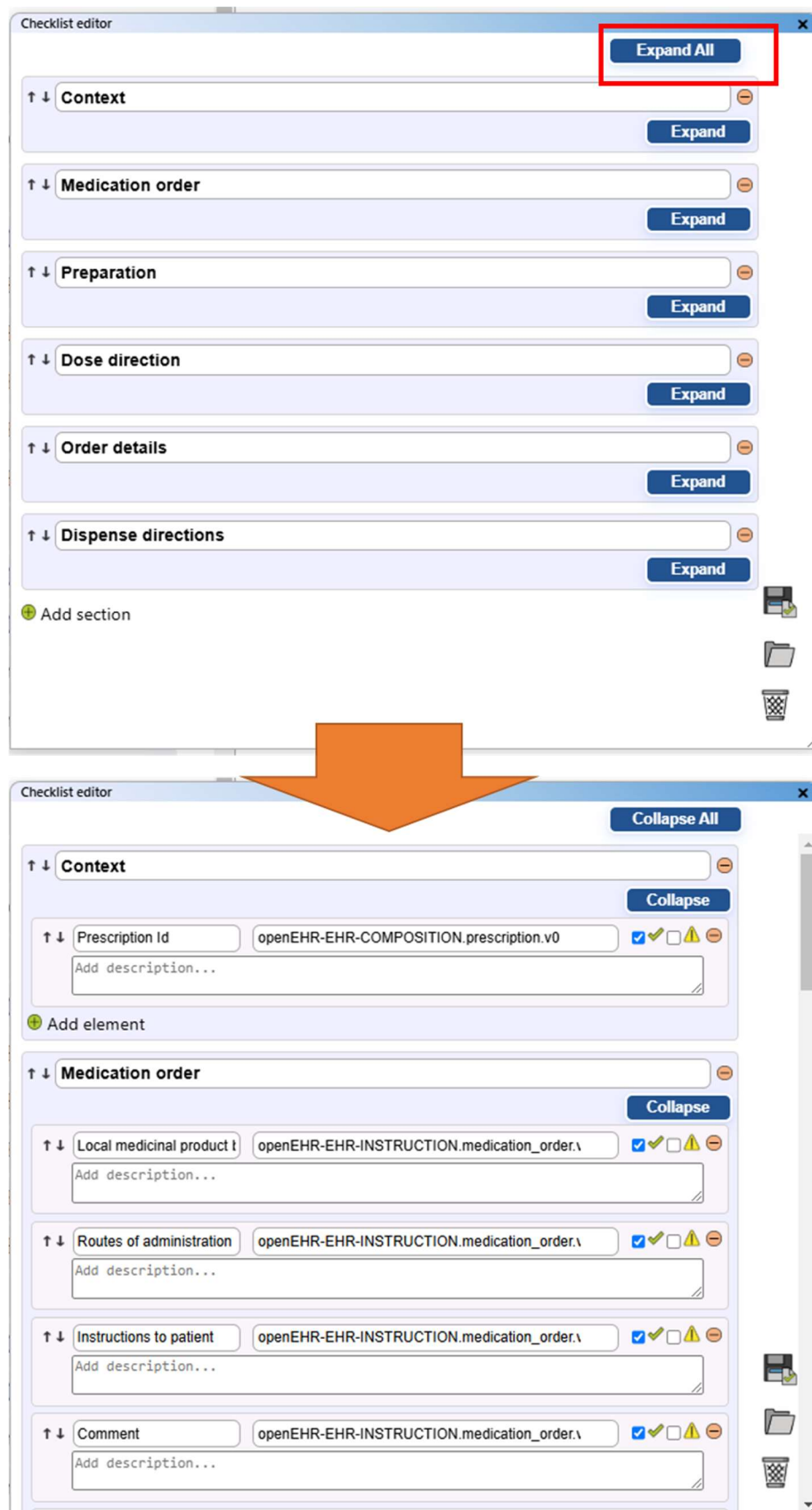
Rename the element with the corresponding input field.

Add element description: Under any element you can add a description in the text field. This description helps to better define the element.



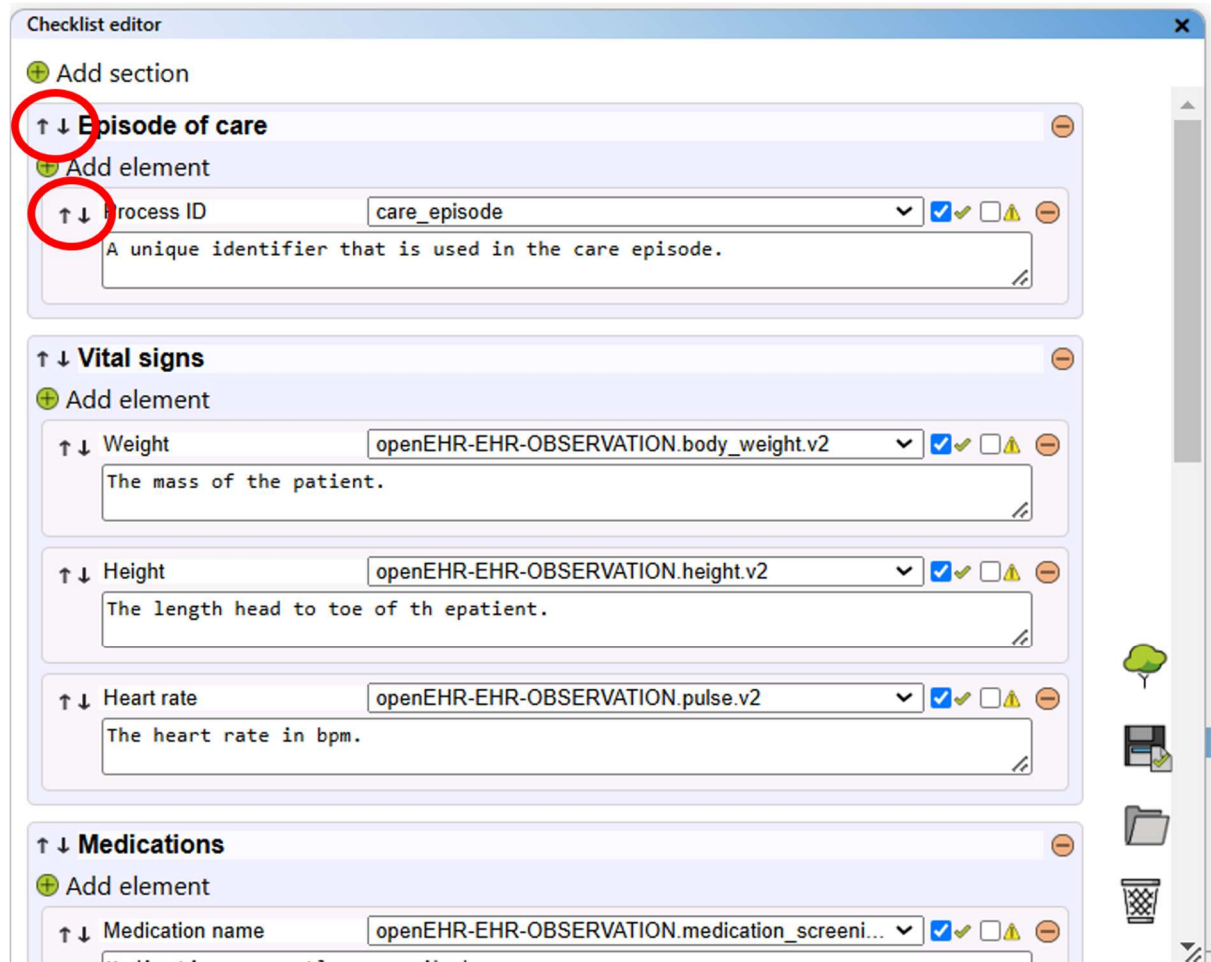
Add a description to an element.

Collapsing/Expanding Branches: Working on the checklist can be challenging. To help you focus on the section you are working on, you can collapse or expand sections as needed. Use the "Collapse All" button at the top to close all sections, or the "Expand All" button to open them. Each section also has a button on its right, labeled "Collapse" or "Expand", depending on its current state. Clicking this button will collapse or expand that specific section.



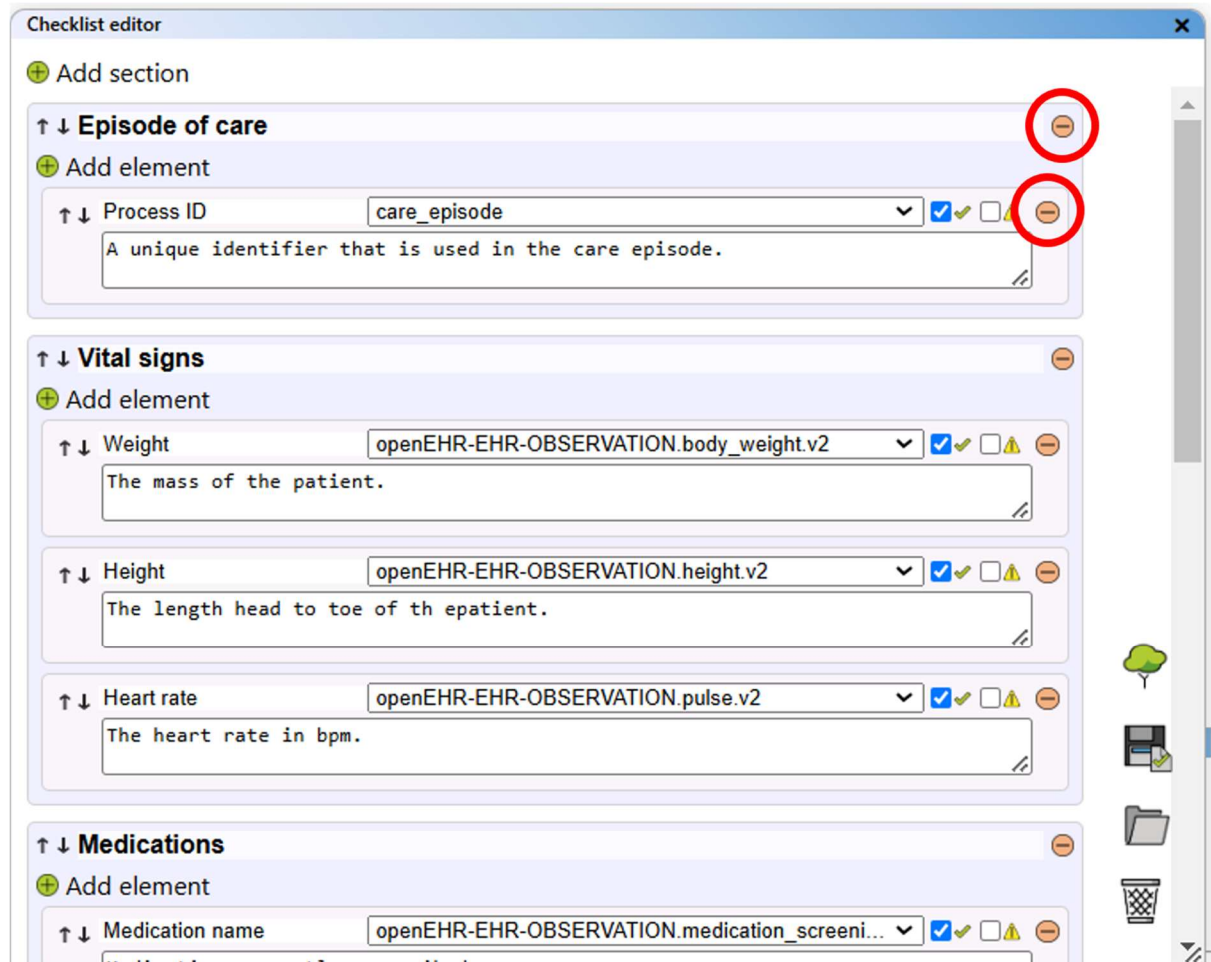
Collapsing the checklist items in the editor.

Move Sections or Elements: If a section or element is not in the desired position, use the arrow icons in front of the name to move it up or down.



Use the up and down arrow to move whole sections or single elements.

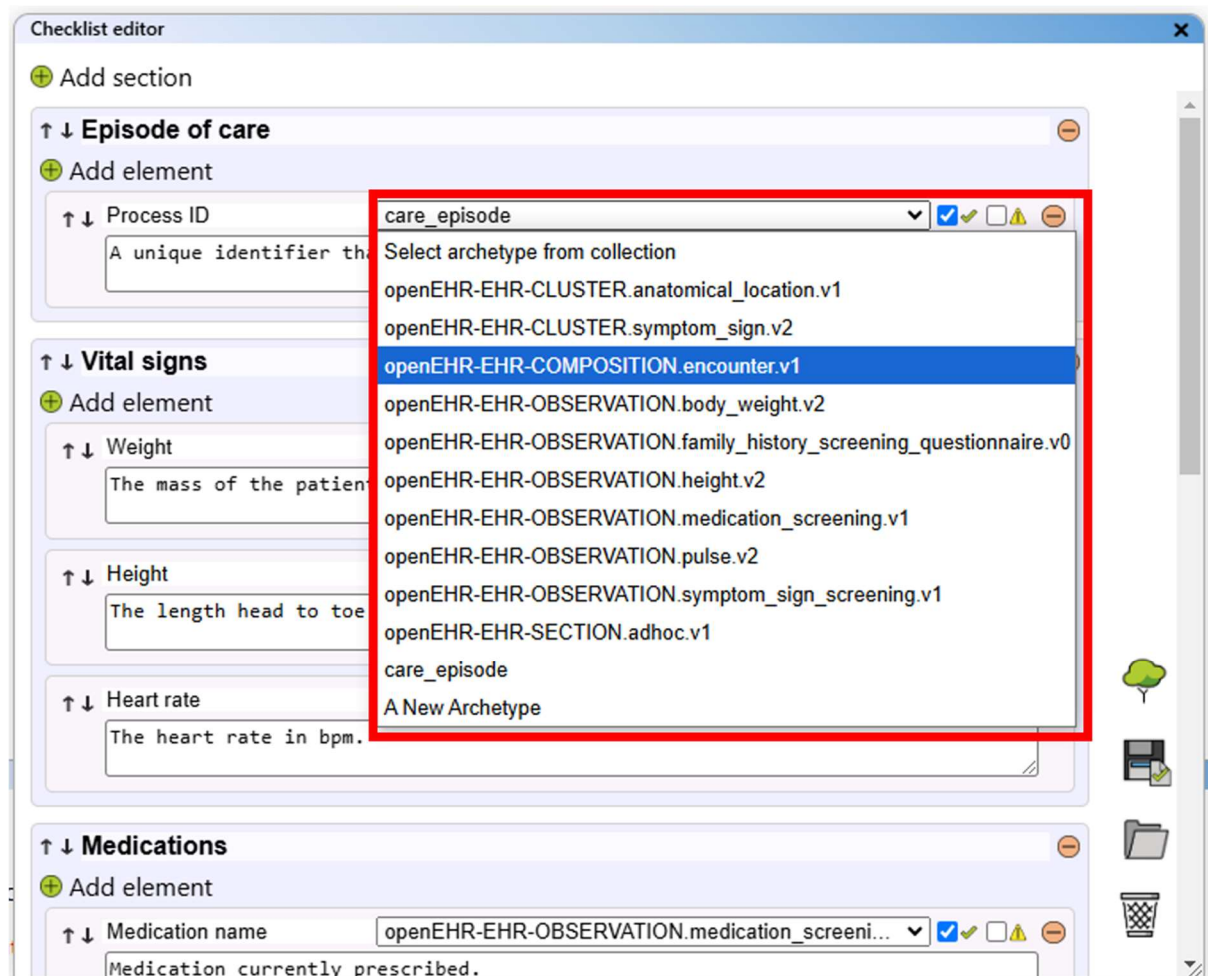
Delete Elements or Sections: To remove an item, click the "Delete" button next to it. Deleting a section will also delete all elements within that section.



Delete whole sections or single elements.

Assigning Archetypes to Data Elements

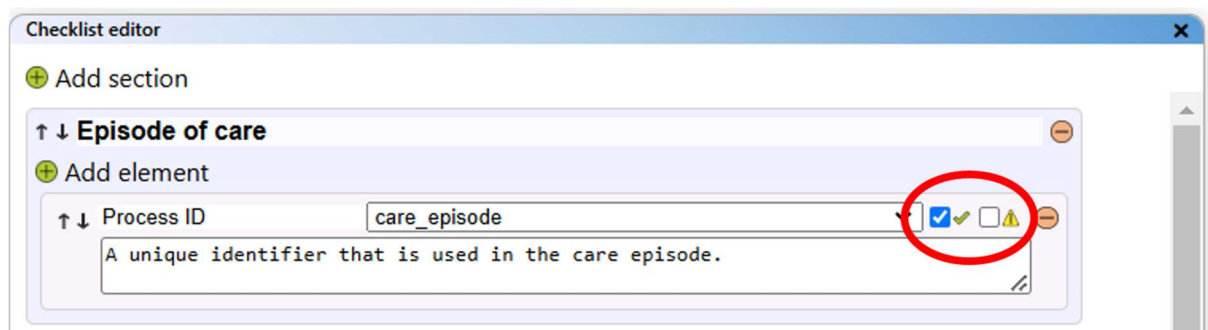
If archetypes are available in the collection, you can assign any element to an archetype, indicating that the archetype contains an item equivalent to your data element. Use the dropdown list next to the element name to select the archetype. If an archetype is removed from the collection, any assignments to it will be reset.



Use the dropdown menu to assign an archetype from the collection to an element.

Flagging Data Elements

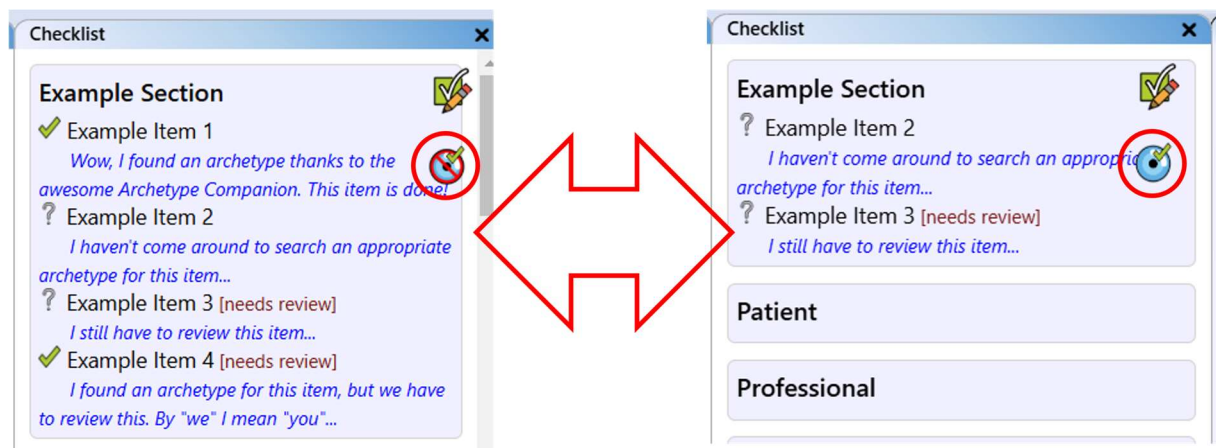
Each element in the checklist editor has two checkboxes. Use the left checkbox to indicate that an element is complete. Use the right checkbox to mark elements that need further review.



Flag elements as approved and/or pending review.

Hide/Show "approved" checklist items

When the checklist is show as a reminder in the right sidebar, you can toggle the visibility of already "approved" items. This makes it easier to see which items are still missing an archetype assignment.



Toggle the visibility of checked items.

Save Checklist to Excel

You can export your checklist as an Excel file for offline editing or sharing. After entering the editor mode, click the "Save Checklist to Excel" button. The file is saved automatically to your browser's download directory. The filename includes the time, date, and project name (which may be modified for compatibility).



Save checklist to an Excel file.

Open Checklist from Excel File

If you prefer editing the checklist in Excel or have a checklist saved in Excel format, you can import it. In the checklist editor, click "Load checklist from Excel". Use the standard file dialogue to select and open your Excel file, which will load its content into the checklist editor.



Load a checklist from an Excel file.

Delete a Checklist from Project

To delete the entire checklist, use the "Delete Checklist" button in the checklist editor.

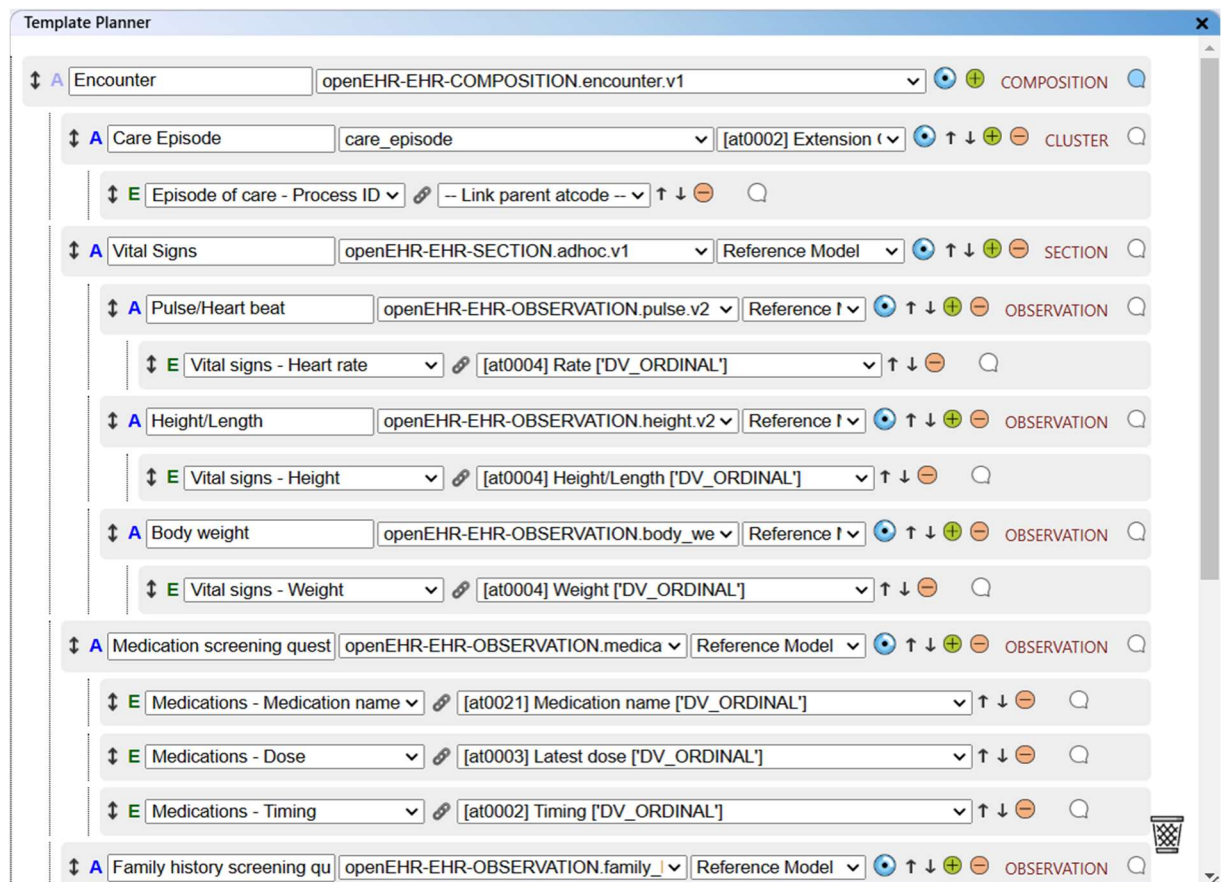


Delete the checklist.

16.11. Managing a Template-like Structure (Template Planner)

The Archetype Companion includes a module that allows you to plan a tree structure, similar to an openEHR template. Please note that this tool does not create actual openEHR templates; instead, it serves as a support tool for designing structures before you begin modelling in a dedicated openEHR modelling tool.

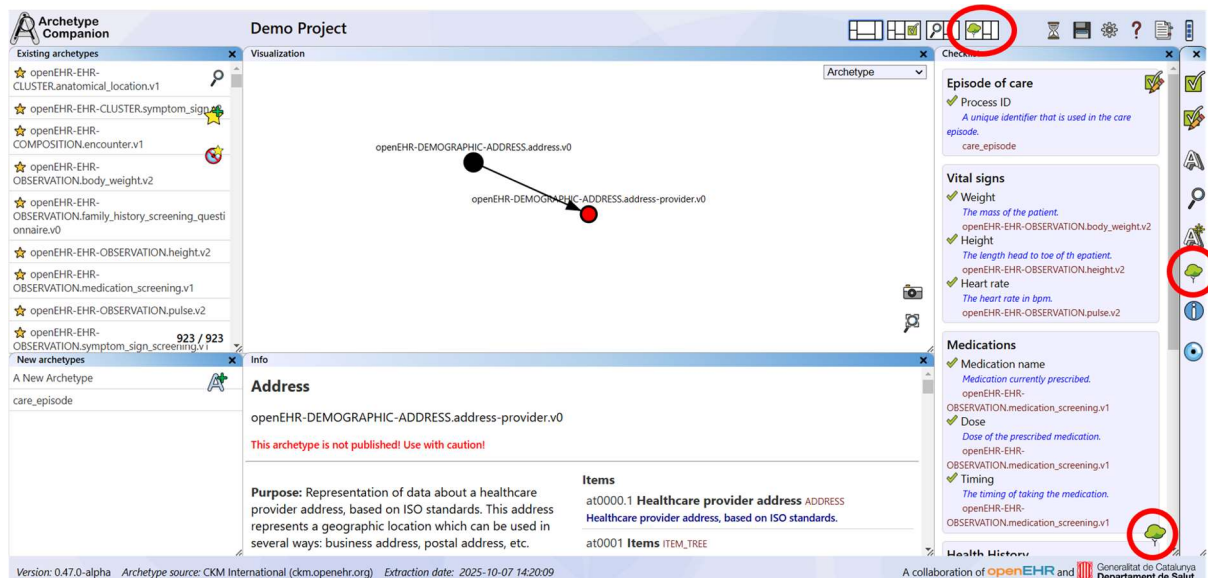
When opened, the Template Planner appears on the left-hand side of the interface.



The open Template Planner.

Where to Find the Template Planner Editor

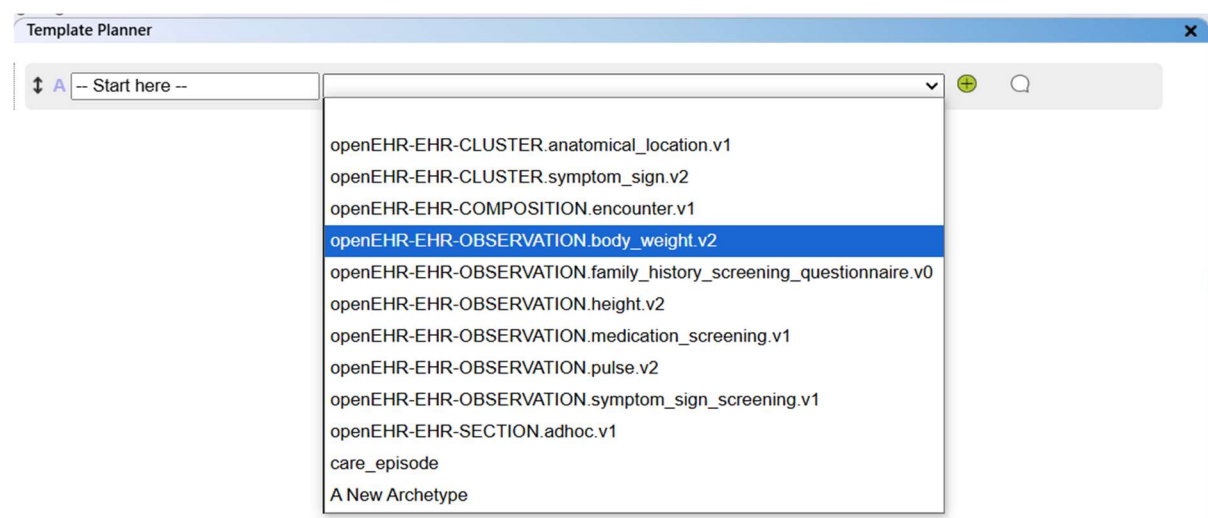
There are several ways to open the Template Planner. To access the standard Template Planner edit view, click the icon located in the upper right-hand corner of the interface. Alternatively, you can open the Template Planner in a floating window by clicking the tree icon in either the checklist view or the checklist editor, or by using the button in the sidebar.



Access the Template Planner via the tree icon.

Selecting an Archetype

When you first open the Template Planner editor, only a single node of the tree is displayed. This node must be an archetype that serves as the root of the "template" (typically a COMPOSITION archetype). You can select one of the archetypes from the dropdown menu; note that only archetypes from your collection will appear here. Once selected, the default concept name of the archetype is inserted as the name. You can change this name if you wish.



Select an archetype from your collection as a node of the tree.

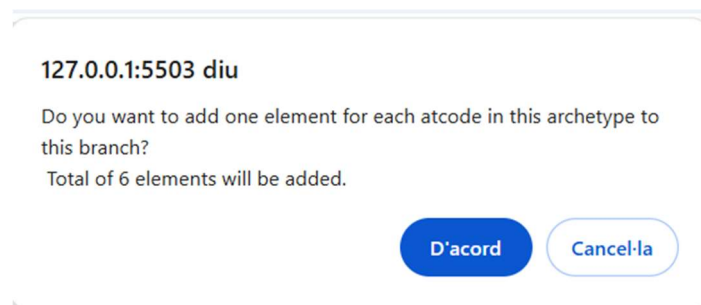
Add archetype elements automatically

On top of the Template Planner editor you find a checkbox "Prompt archetype autofill".



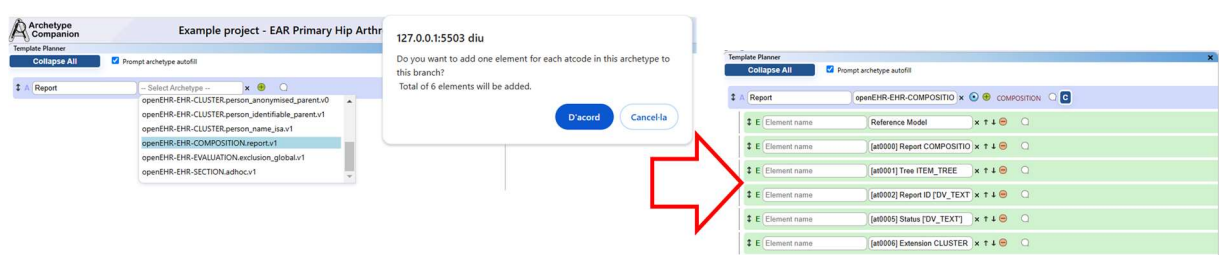
Enable or disable the autofill option.

When selecting an archetype and the autofill lfeature is enabled, a message pops up that asks for confirmation of the autofill.



Confirm autofill on archetype selection.

If the message is confirmed, all elements of the archetype are added to the current branch.

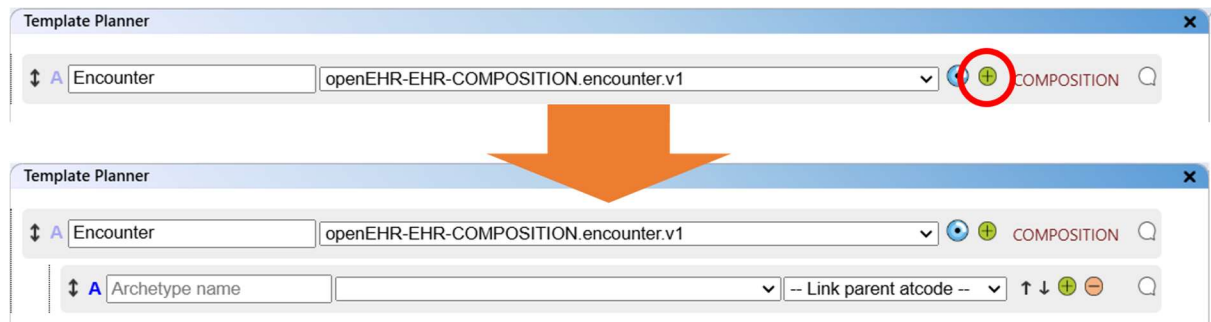


Elements are added automatically to the archetype branch.

Adding a Branch

Each archetype node in the tree can have subsequent archetype or element nodes associated with it. For example, a SECTION archetype can include an OBSERVATION archetype, which in turn contains a data element. Note that an element cannot contain other elements or archetypes.

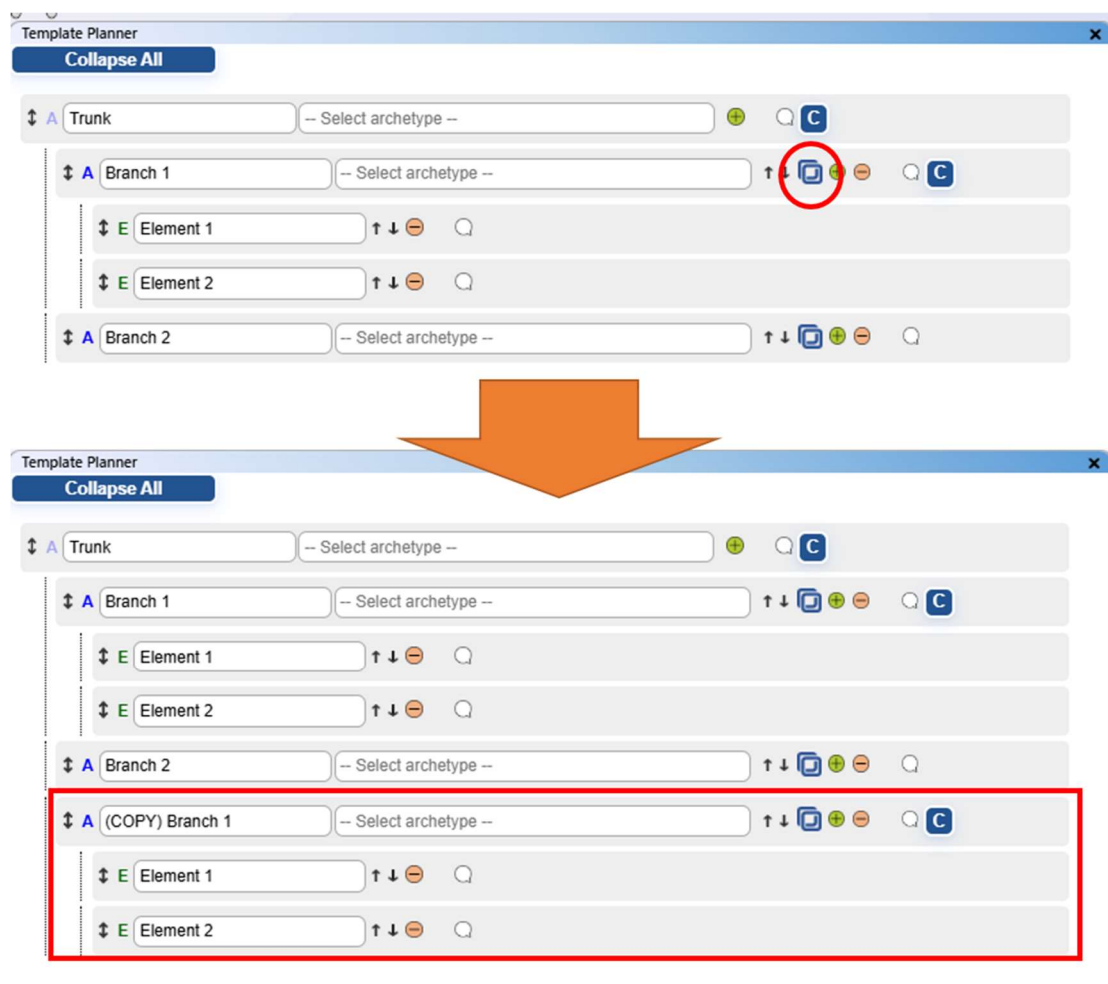
To create a branch under an archetype node, click the "+" icon to the right of the node. This will create a new archetype node.



Adding a branch node to the tree.

Cloning a Branch

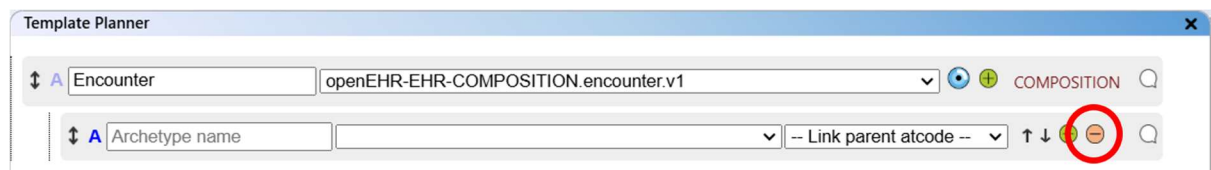
If you need to model structures with similar sub-structures, you can easily duplicate an entire branch—including its archetype and element hierarchy—by clicking the "Clone branch" button on the relevant archetype node.



Cloning a branch node of the tree.

Deleting a Node or a Branch

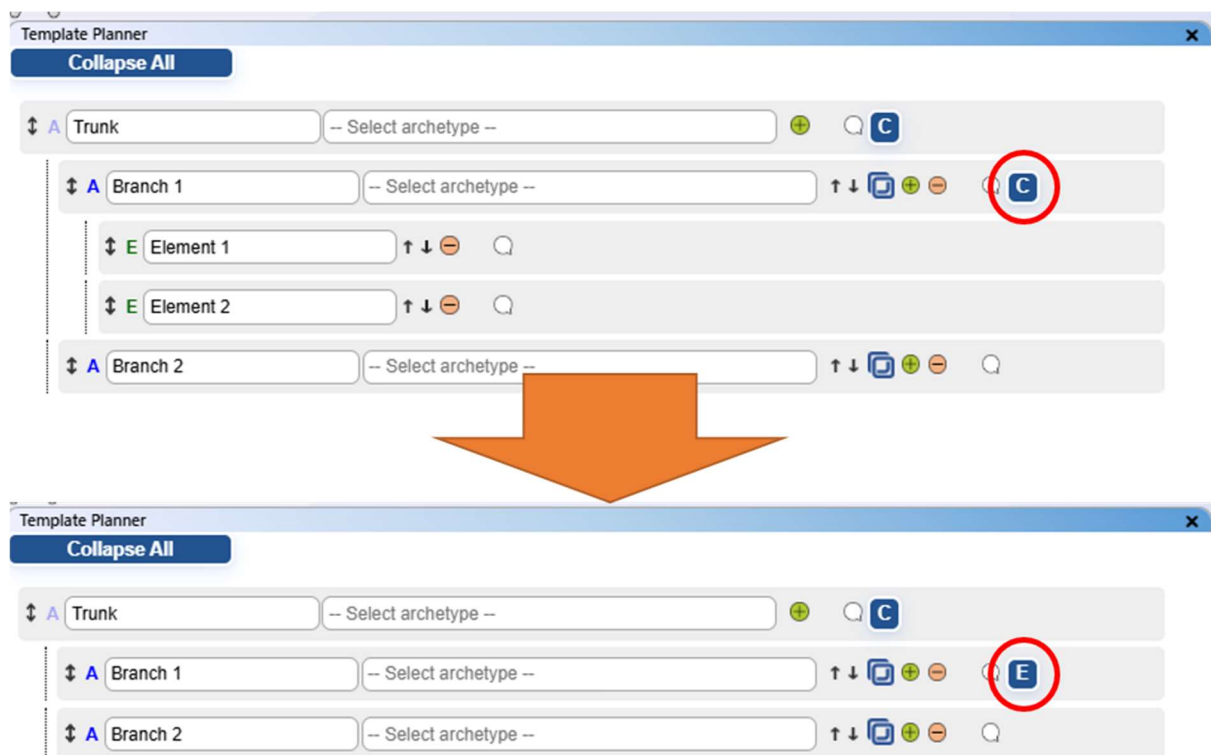
To delete a node from the tree, click the "-" icon to the right of the node. This will remove the node, along with any underlying nodes associated with it.



Deleting a node from the tree.

Collapsing/Expanding Branches

Navigating complex tree-like structures can be challenging, and it's easy to lose track of the overall layout. To help you manage these structures more efficiently, you can collapse or expand branches as needed. Use the "Collapse All" button at the top to close all branches, or the "Expand All" button to open them. Each archetype node with sub-branches also has a button on its right, labeled "C" (for collapse) or "E" (for expand), depending on its current state. Clicking this button will collapse or expand that specific branch.

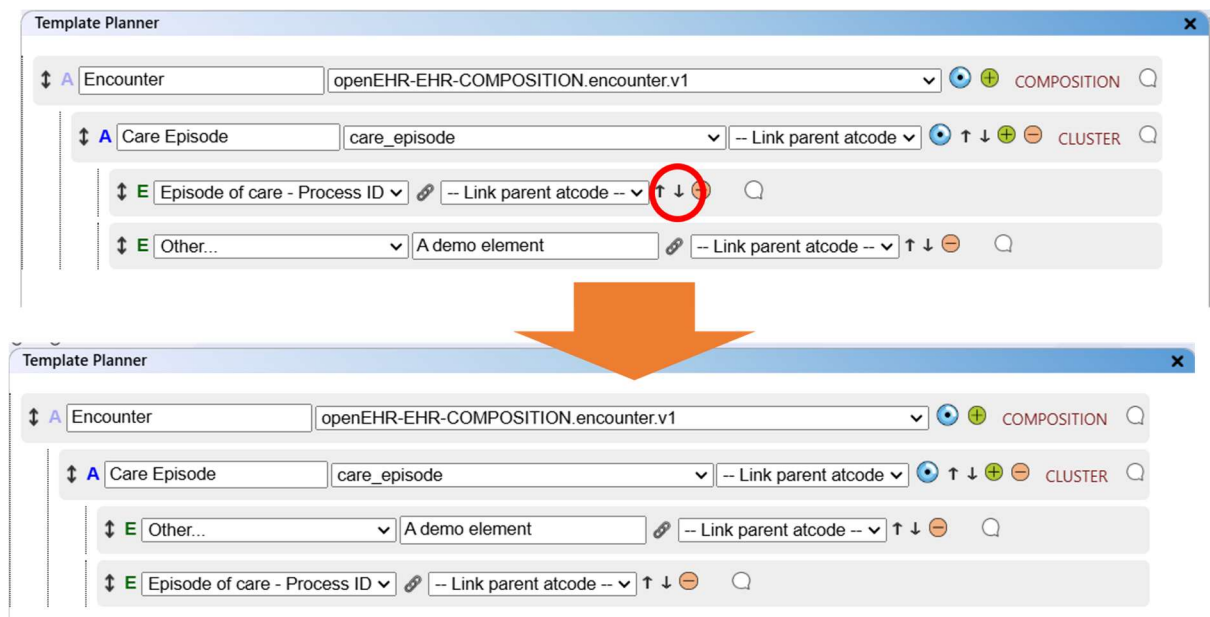


Collapsing a node from the tree.

Moving Nodes and Branches Around

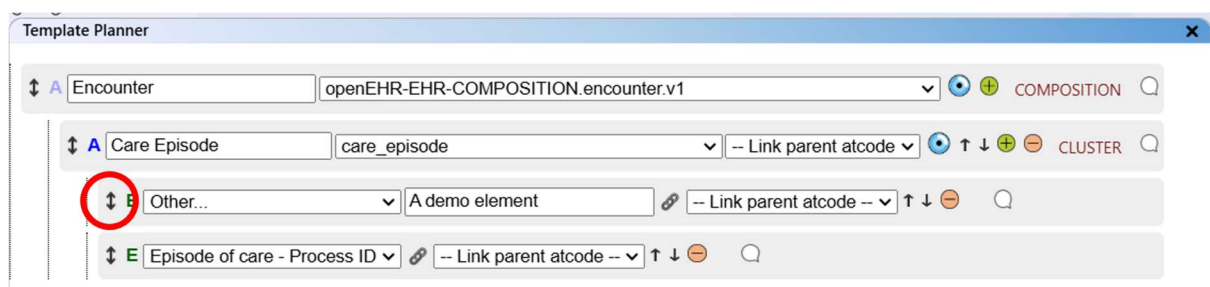
If the order of the nodes or branches needs to be changed in the planned structure, the individual nodes can be moved.

Every node has an up and down arrow on the right side. By clicking on these buttons, the individual node is moved up or down.



Move a node up or down.

Whole branches and elements can be moved via drag and drop. Click and hold on the double-arrow on the left side of the node. You can now drag the node into another node.



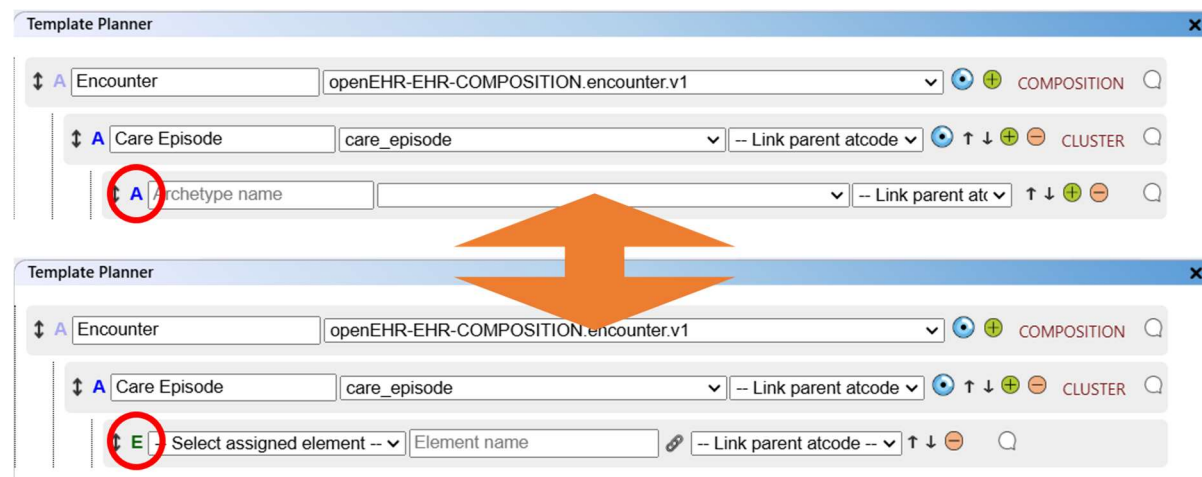
Drag and drop a node or branch.

Switching Between Archetype and Data Element

A node in this tree can represent either an archetype or a data element.

Once a node is created, you can switch between the two modes by clicking the "A" (or "E") icon to the left of the node.

Please note that changing modes will delete the information associated with the node (such as the name or archetype ID). Additionally, if you change an archetype node that has sub-nodes, all sub-nodes will also be deleted.

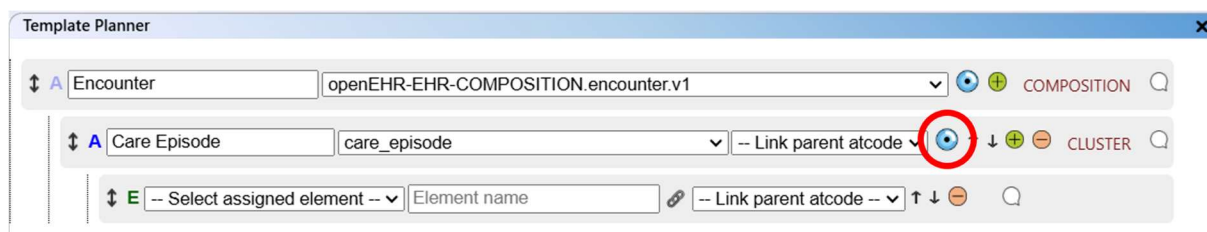


Switching between "archetype" and "data element" node modes.

Archetype Nodes

In contrast to element nodes, an archetype node has a selection dropdown menu to select the archetype from the "archetype collection".

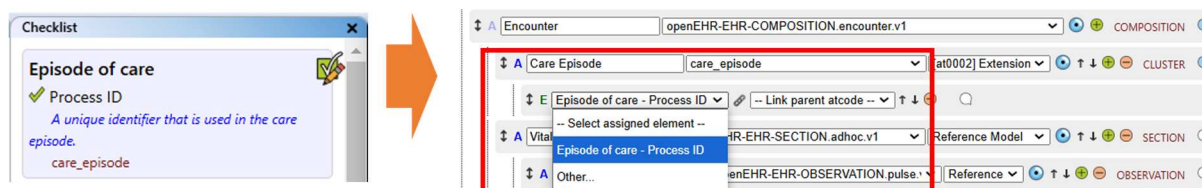
One special item in an archetype node is the ability to "focus" on this node. If an archetype is selected from the collection, you can click on the focus icon. This will select the archetype and load its information in the archetype information window and the visualisation window.



The focus on node icon.

Data Element Nodes

Data element nodes come in two forms. If an archetype has data elements assigned in the checklist, the data element node in the tree that has been added under this archetype becomes a selection. You can select one of the assigned data elements or enter another data element name. Note that the data element names consist of the section and the element name of the checklist.



If an archetype has data elements assigned in the checklist, the data element node in the tree is a

selection.

If an archetype has no data element assigned in the checklist, the data element node in the tree is an input field for the name.



If an archetype has no data element assigned in the checklist, the data element node in the tree is an input field for the name.

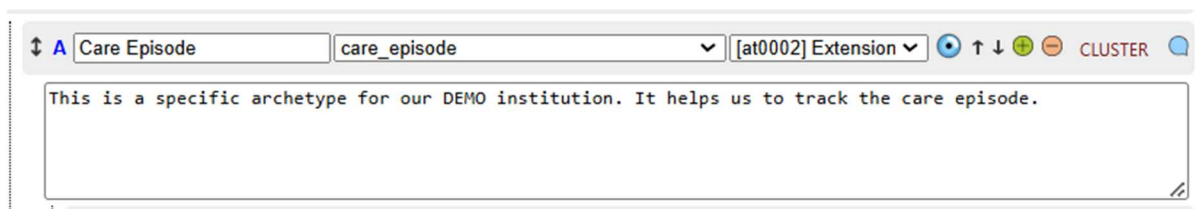
Commenting Nodes

Every node in the tree can have a comment. To open the comment text field, you can click on the speech bubble on the right of the node.



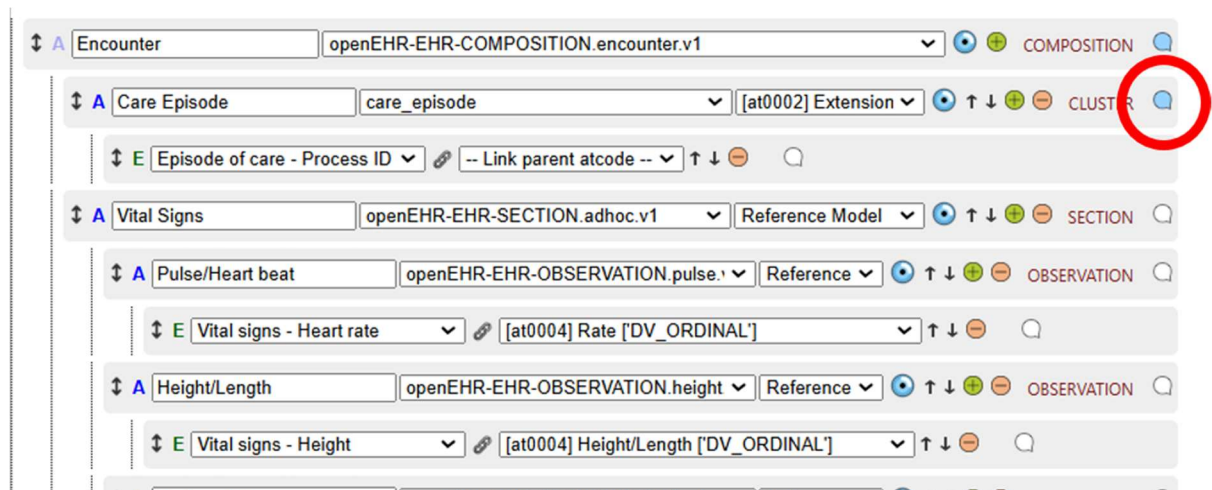
Toggle the comment text field for a node.

This opens a text field in which you can store a comment. You can collapse the text field by clicking again on the speech bubble.



Enter a comment into the text field.

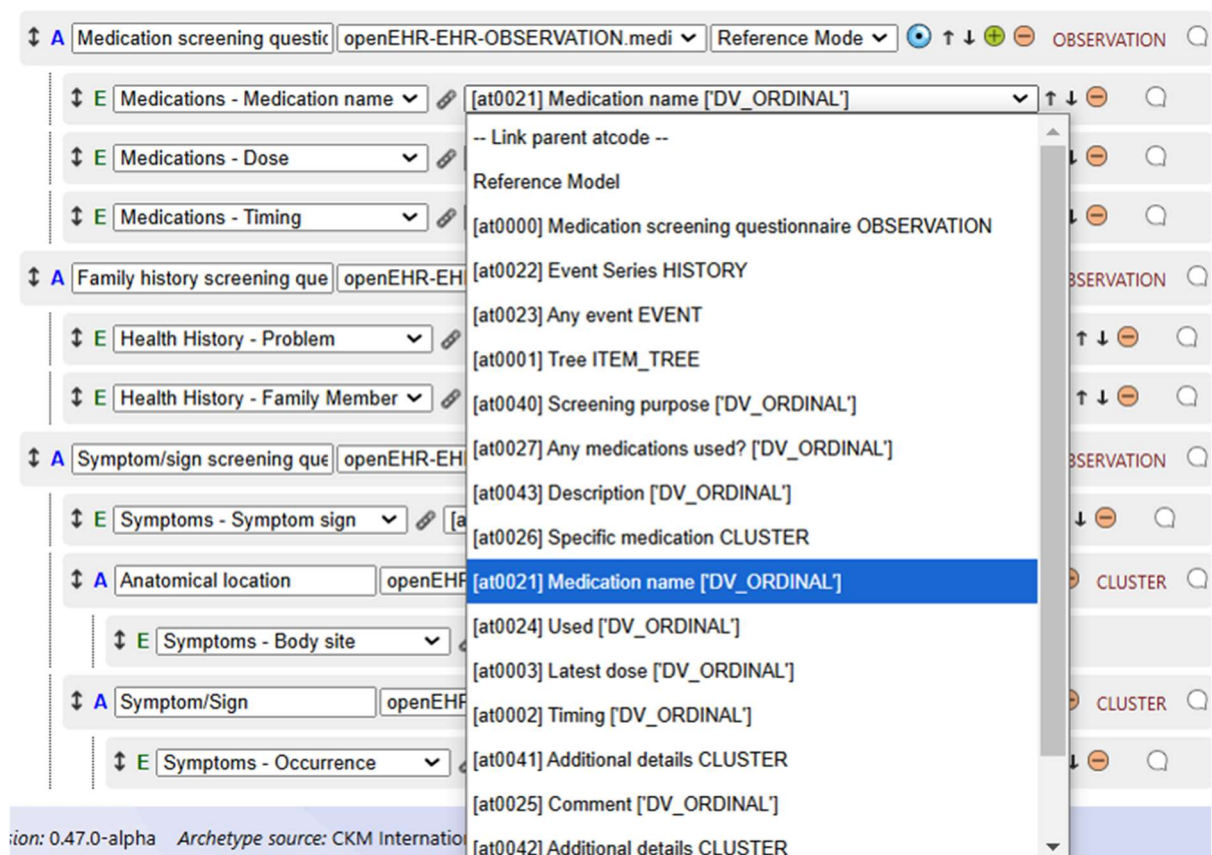
Note that in the presence of a comment, the speech bubble turns blue. If the comment text fields are collapsed, it is easy to see where comments exist.



Blue speech bubbles indicate the presence of a comment.

Assigning "atcode" Item to Archetype or Element

If a node is added to an archetype node that has been selected from the archetype collection, you can select the atcode of the archetype that you would like to add the new node to. You can select the code from the dropdown menu. The assignment of the code will be visible in the project report.

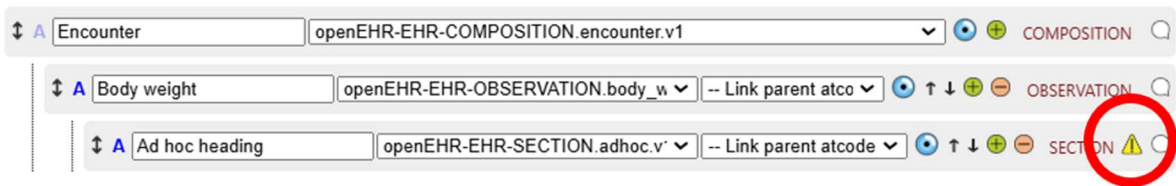


Assign atcode of the parent archetype to the node.

Compatibility Warnings

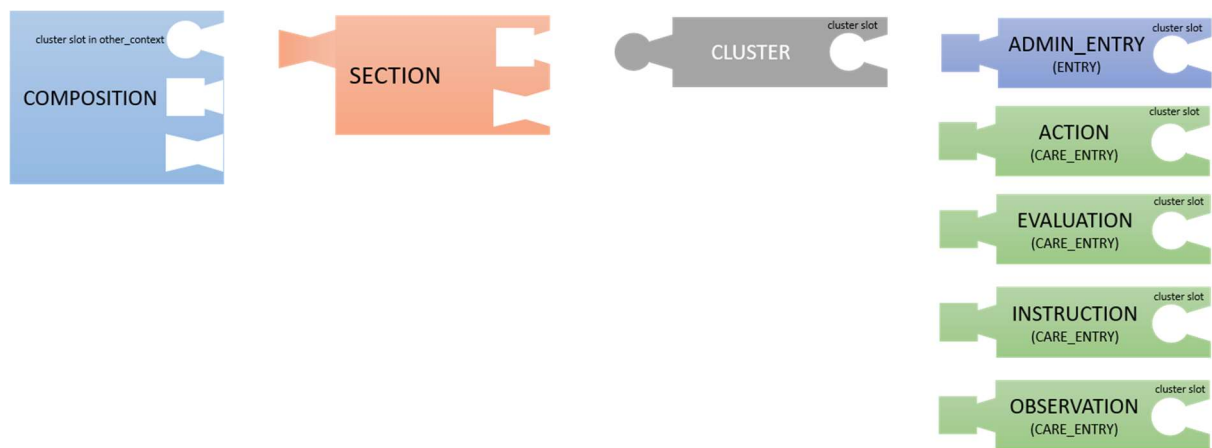
As a support for the openEHR template modelling process, the Template Planner offers a compatibility warning for the different archetype classes. In the openEHR framework, not every class can be incorporated into any other class. A possible compatibility issue is indicated by a yellow warning sign to the left of the archetype node.

Note that this is only a reminder to check the compatibility between different archetypes. A warning can be shown, even if the archetypes are compatible.



Compatibility warning for an archetype class that has been included (possibly) incorrectly into another archetype class.

The compatibility between archetype classes can be visualised as puzzle pieces:



Compatibility between archetype classes.

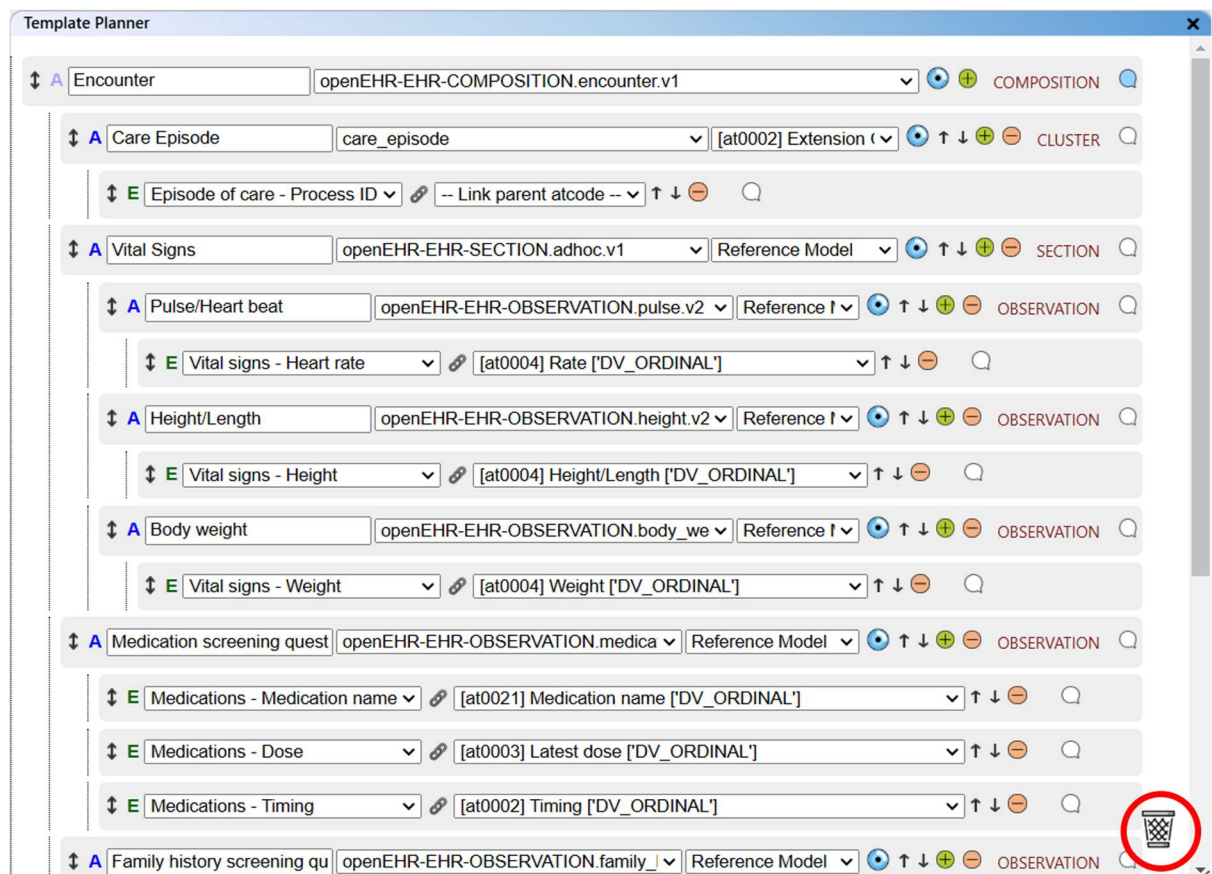
Translating the previous figure into a table, we can easily check if a class is compatible or not.

Class	Can be added to	Can contain
COMPOSITION	---	CLUSTER (if slot added to context/other_context) SECTION ENTRY

Class	Can be added to	Can contain
SECTION	COMPOSITION SECTION	ENTRY SECTION
CLUSTER	COMPOSITION (if slot provided) CLUSTER (if slot provided) ENTRY (if slot provided)	CLUSTER (if slot provided)
ENTRY (ADMIN_ENTRY, ACTION, EVALUATION, INSTRUCTION, OBSERVATION)	COMPOSITION SECTION	CLUSTER (if slot provided)

Delete All Template Planner Data

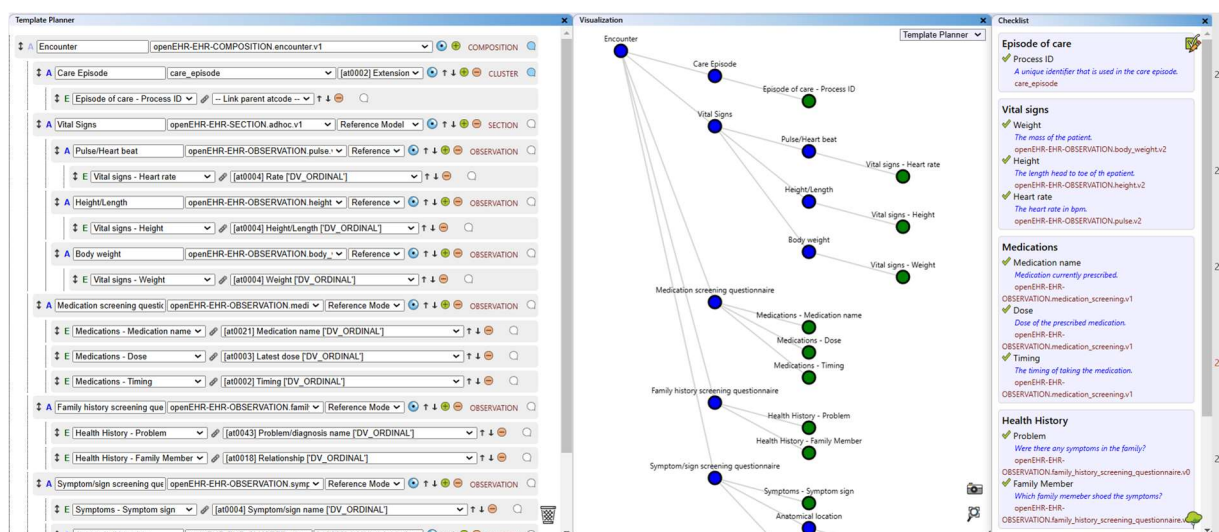
To delete all data from the Template Planner, click the "Delete" icon in the lower right-hand corner of the window.



Delete everything in the tree.

Template Planner and Visualisation

If you want to see a simple representation of the Template Planner content, you can select "Template Planner" in the visualisation window. The nodes of this visualisation show the archetypes (blue) and assigned elements (green). This view can help you to navigate the planned structure more easily, without the extended information of the Template Planner.



Network visualisation of the planned tree structure.

Template Planner Data in the Report

When creating the report for the project, the data from the Template Planner is used to create a list with all elements that were defined. An example of this list, as found in the report, can be seen here:

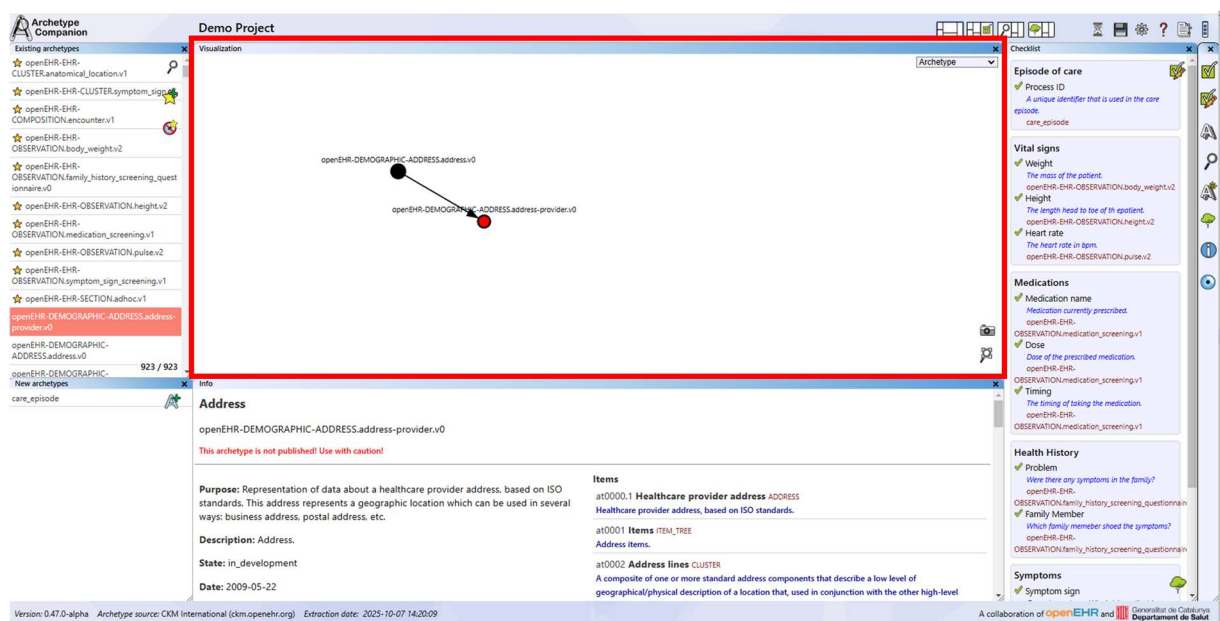
- **Encounter**
openEHR-EHR-COMPOSITION.encounter.v1
Type: COMPOSITION
archetype
I think that this is the best archetype for the job.
 - **Care Episode** -- [at0002] Extension CLUSTER
care_episode
Type: CLUSTER
archetype
This is a specific archetype for our DEMO institution. It helps us to track the care episode.
 - **Episode of care - Process ID**
element
 - **Vital Signs** -- Reference Model
openEHR-EHR-SECTION.adhoc.v1
Type: SECTION
archetype
 - **Pulse/Heart beat** -- Reference Model
openEHR-EHR-OBSERVATION.pulse.v2
Type: OBSERVATION
archetype
 - **Vital signs - Heart rate** -- [at0004] Rate ['DV_ORDINAL']
element
 - **Height/Length** -- Reference Model
openEHR-EHR-OBSERVATION.height.v2
Type: OBSERVATION
archetype
 - **Vital signs - Height** -- [at0004] Height/Length ['DV_ORDINAL']
element
 - **Body weight** -- Reference Model
openEHR-EHR-OBSERVATION.body_weight.v2
Type: OBSERVATION
archetype
 - **Vital signs - Weight** -- [at0004] Weight ['DV_ORDINAL']
element

16.12. Visualisation

The network visualisation feature of the application provides an intuitive way to see relationships between archetypes. Different visualisation modes allow you to display archetype dependencies, similarities, your archetype collection, the overall project structure, and the template planner tree.

Where to Find the Visualisation Window

In the standard main view, the visualisation feature is located in the centre section of the screen. Before exploring the different views, it is helpful to understand how to navigate the visualisation canvas.



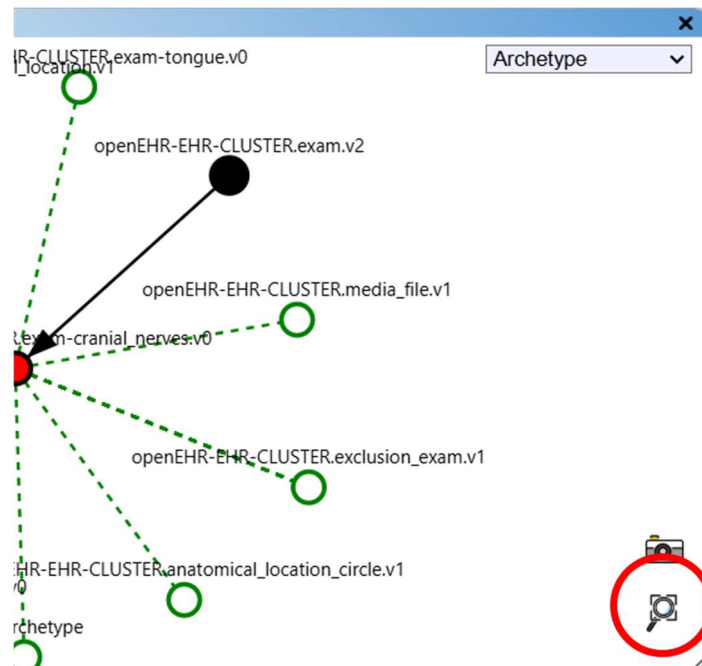
Where to find the visualisation.

Navigating the Visualisation

Move: To move the network visualisation canvas, click and hold the left mouse button on an empty space and drag the network around.

Zoom: To zoom in or out, use your mouse wheel while the cursor is over the canvas. Alternatively, you can use pinch gestures on a trackpad.

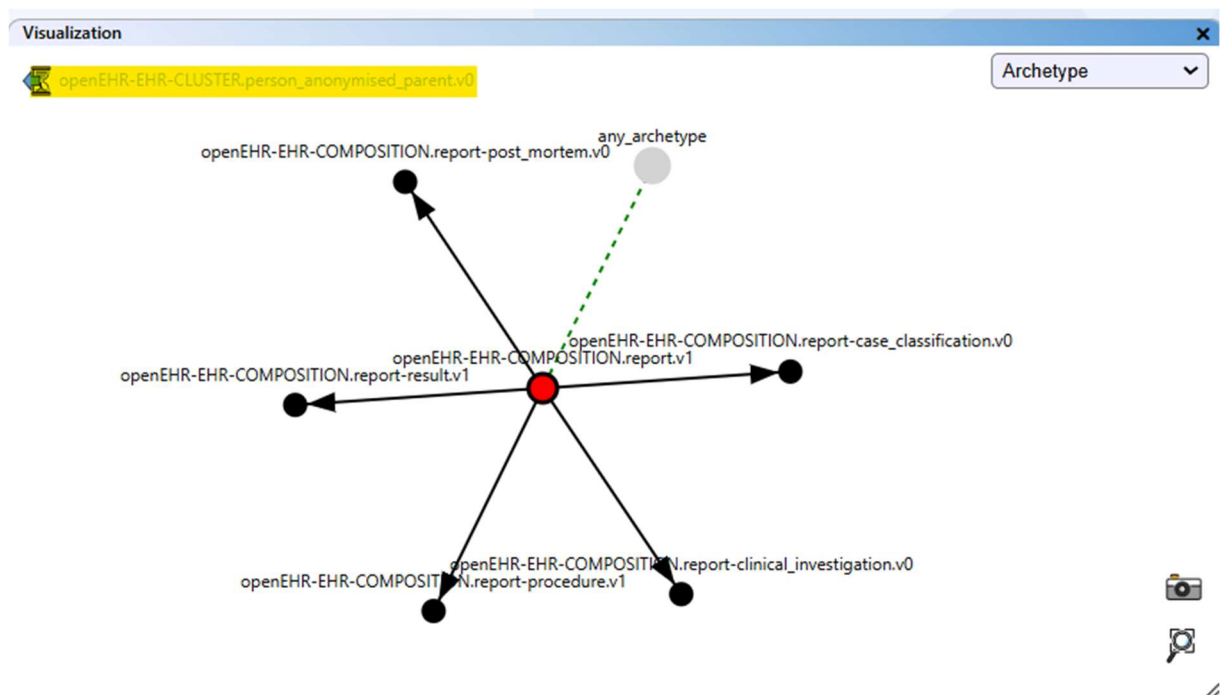
Reset Zoom: To reset the position and zoom, click the "Reset Zoom" icon in the lower right-hand corner of the visualisation.



Reset zoom button.

Focus on Archetype: To focus on a specific archetype in the network, click on its node with the left mouse button.

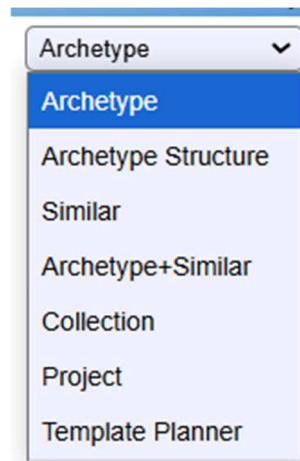
Go back one archetype at the time: When more than one archetype has been shown in the past, a button appears in the upper left corner. By clicking this button, you can focus on the last visited archetype.



One step back button.

The Different Views

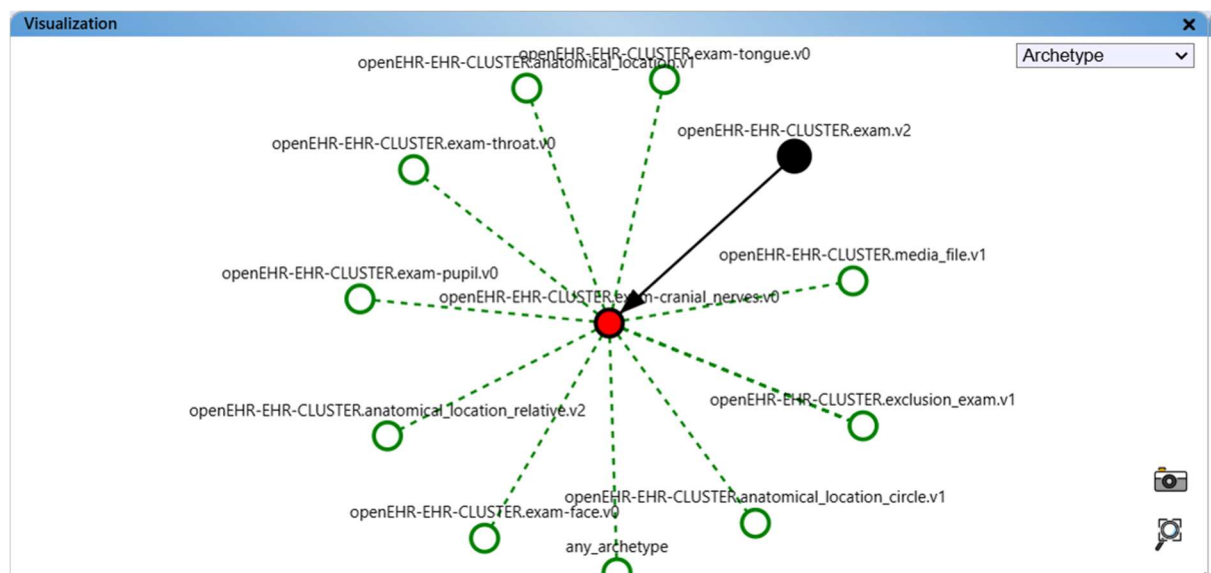
The network visualisation has various views that can be selected with the dropdown menu on the upper right side of the window. You can select between *Archetype*, *Archetype Structure*, *Similar*, *Archetype + Similar*, *Collection*, *Project*, and *Template Planner* view.



The view selection.

Archetype:

- In the "Archetype" view, you can see the relationships of an archetype with others.
- The currently selected archetype is shown as a red node.
- Parent-child relationships are indicated by solid black lines, with arrows pointing from parent to child.
- Inclusion and exclusion relationships are shown as green and red dashed lines, respectively.
- This is the default view when opening the application.



Archetype Structure:

-

Similar:

- Visualization

Similar

openEHR-EHR-CLUSTER.exam-lens.v0

openEHR-EHR-CLUSTER.exam-anterior_chamber_eye.v0

openEHR-EHR-CLUSTER.exam-fundus_eye.v0

openEHR-EHR-CLUSTER.exam-middle_ear.v0

openEHR-EHR-CLUSTER.exam-thyroid.v0

openEHR-EHR-CLUSTER.exam-cranial_nerves.v0

openEHR-EHR-CLUSTER.exam-toe.v0

openEHR-EHR-CLUSTER.exam-respiratory_system.v0

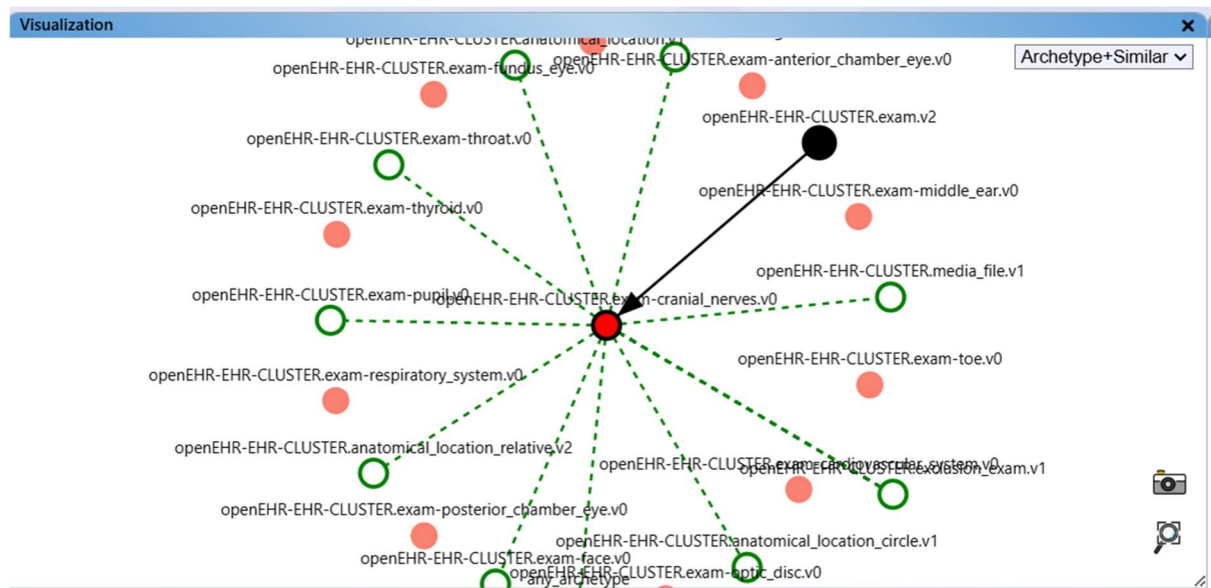
openEHR-EHR-CLUSTER.exam-cardiovascular_system.v0

openEHR-EHR-CLUSTER.exam-posterior_chamber_eye.v0

openEHR-EHR-CLUSTER.exam-optic_disc.v0

Archetype + Similar:

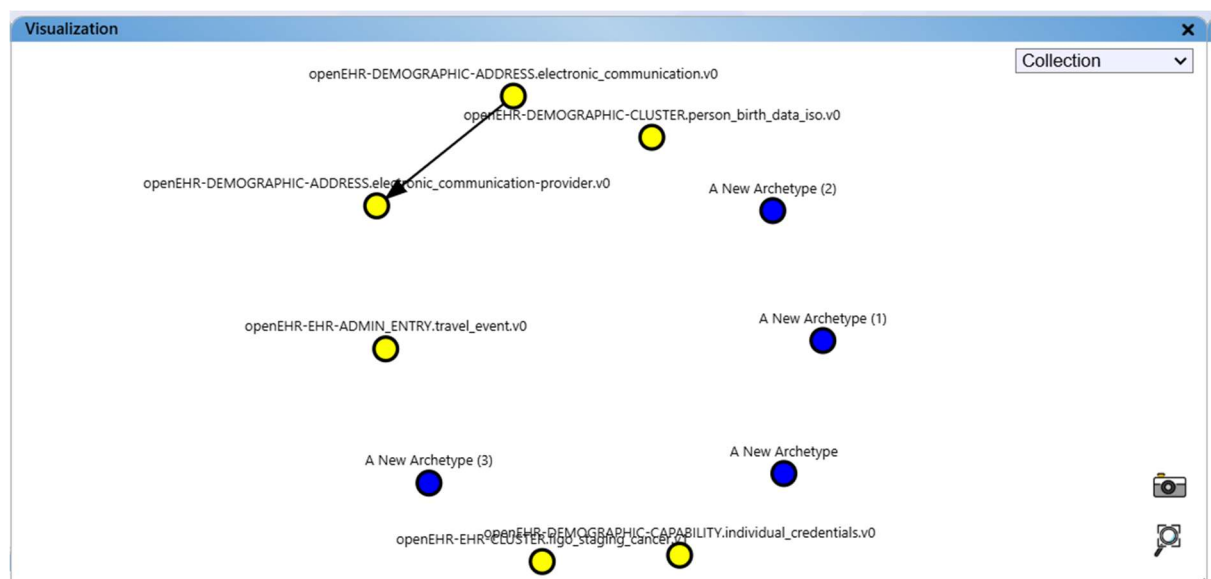
- This view combines the "Archetype" and "Similar" views, showing both relationships and recommendations.



The combined archetypes view.

Collection:

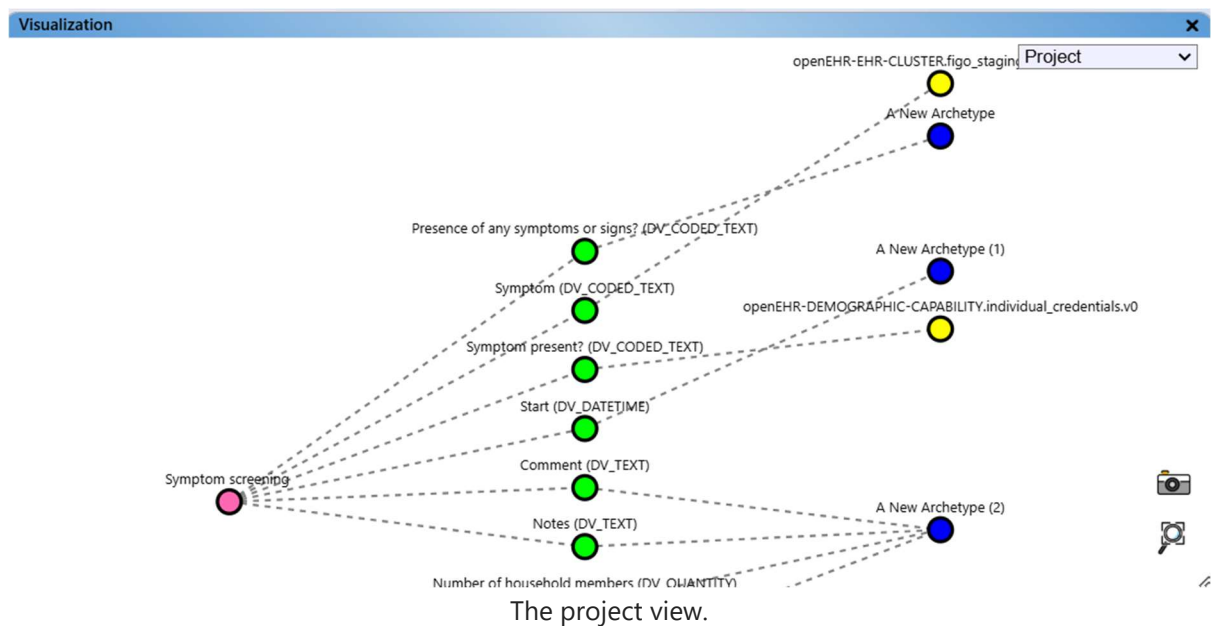
- The "Collection" view shows relationships between archetypes in your collection list.
- There is no red focused archetype.
- Node colours distinguish between archetypes from the database (yellow) and new archetypes (blue).
- Clicking a node selects it and displays its information.
- Relationships are shown with solid black lines (parent-child) and dashed red or green lines (exclusions or inclusions).



The collection view.

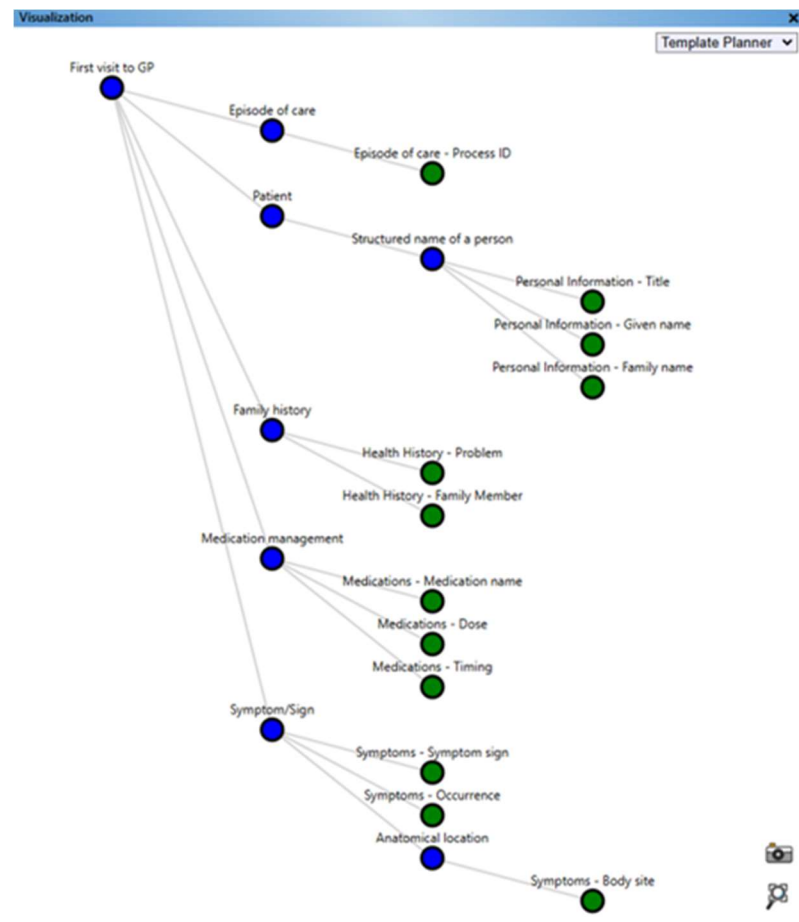
Project:

- The "Project" view displays the assignment of archetypes to data elements defined in your checklist.
- Three columns show section names (pink), data elements (green), and assigned archetypes (blue or yellow).
- Data elements are connected to their sections with grey dashed lines, and assigned archetypes are shown in the third column, connected to data elements with grey dashed lines.
- Node colours again distinguish between database and new archetypes.
- This view provides an overview of all archetypes and checklists in your project, including their dependencies.



Template Planner:

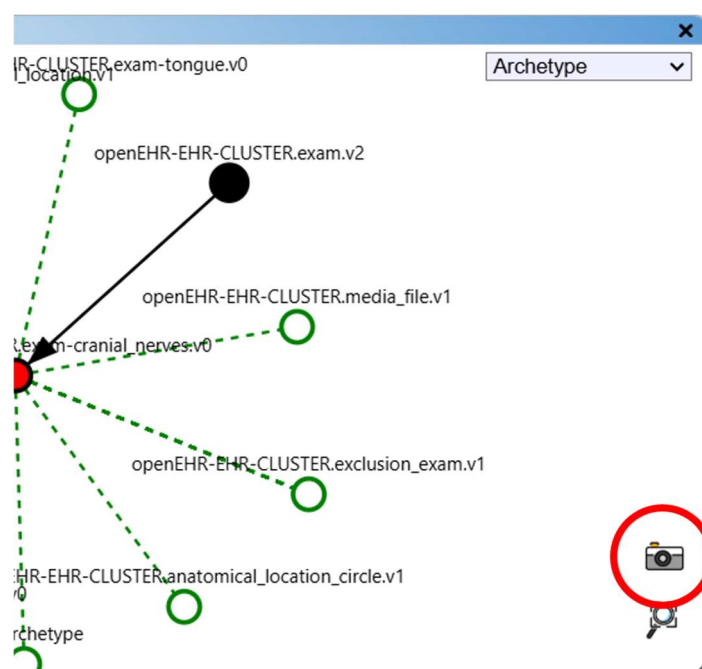
- The "Template Planner" view displays the archetype and data element structure created with the Template Planner module.
- Archetypes are shown as blue nodes, data elements as green nodes.
- The structure of archetypes and data elements is indicated by a grey line.



The Template Planner view.

Export Visualisation View

For documentation, you can export the current visualisation as an SVG image. Click the "Save as SVG image" button in the lower right-hand corner of the visualisation.



Save to SVG file button.

The file is saved automatically to your browser's download directory, with a filename containing the time, date, project name, and view mode. The project name may be modified for compatibility.

16.13. Glossary

The Archetype Companion provides a glossary of openEHR classes. The information on openEHR classes has been extracted previously from the following sources:

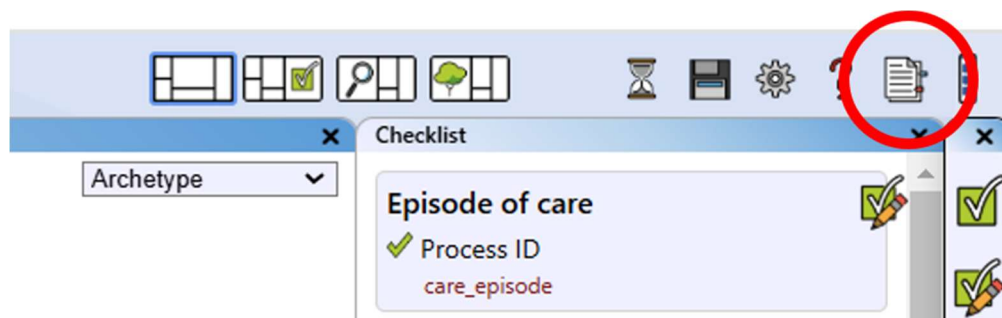
- [BASE Types 1.2.0](#)
- [Reference Model: Common](#)
- [Reference Model: Data Structures](#)
- [Reference Model: Data Types](#)
- [Reference Model: Demographic](#)
- [Reference Model: EHR Extract](#)
- [Reference Model: EHR](#)
- [Reference Model: Integration](#)
- [Reference Model: Support](#)
- [openEHR Class Index](#)

Legal Notice:

This glossary is provided for informational purposes only. We do not claim authorship of the original openEHR content, nor do we assert any copyright over it. All copyrights and intellectual property rights remain with the respective owners and authors. The information is presented as is, without warranty of any kind, either express or implied. Users must ensure that any use of this content complies with the applicable licences and terms set by the original copyright holders. No modifications or redistribution of the original material are implied or encouraged by this resource.

Where to Find the Glossary

The glossary can be accessed easily via the icon in the upper right-hand corner of the interface:

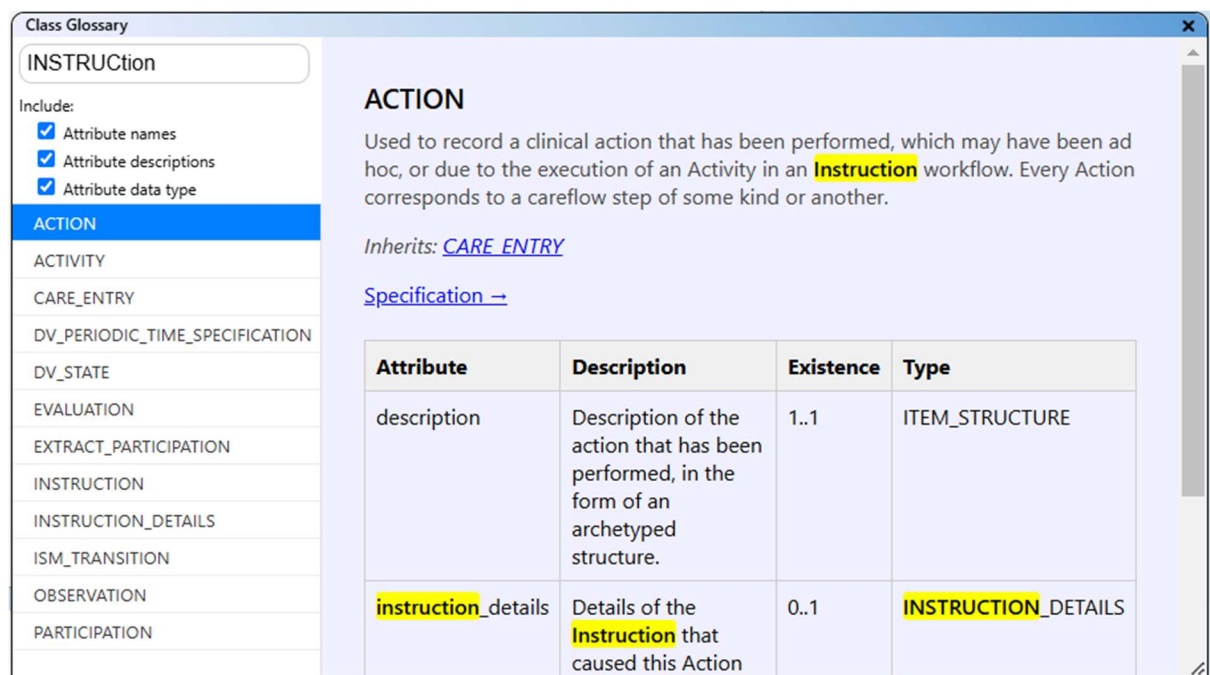


Opening the glossary with the icon.

How to Search the Glossary

In the upper left-hand corner of the window, you will find the search bar. Here you can type a search term. With every keystroke, the search result list beneath is updated. Click on one of the search results to see the corresponding entry.

By default, only the class name and description are searched. You can add attribute names, descriptions and data type to the search by ticking the corresponding checkboxes.



The glossary.

Close the Glossary

You can close the glossary window with the "X" button in the upper right-hand corner of the window.

16.14. Data Storage

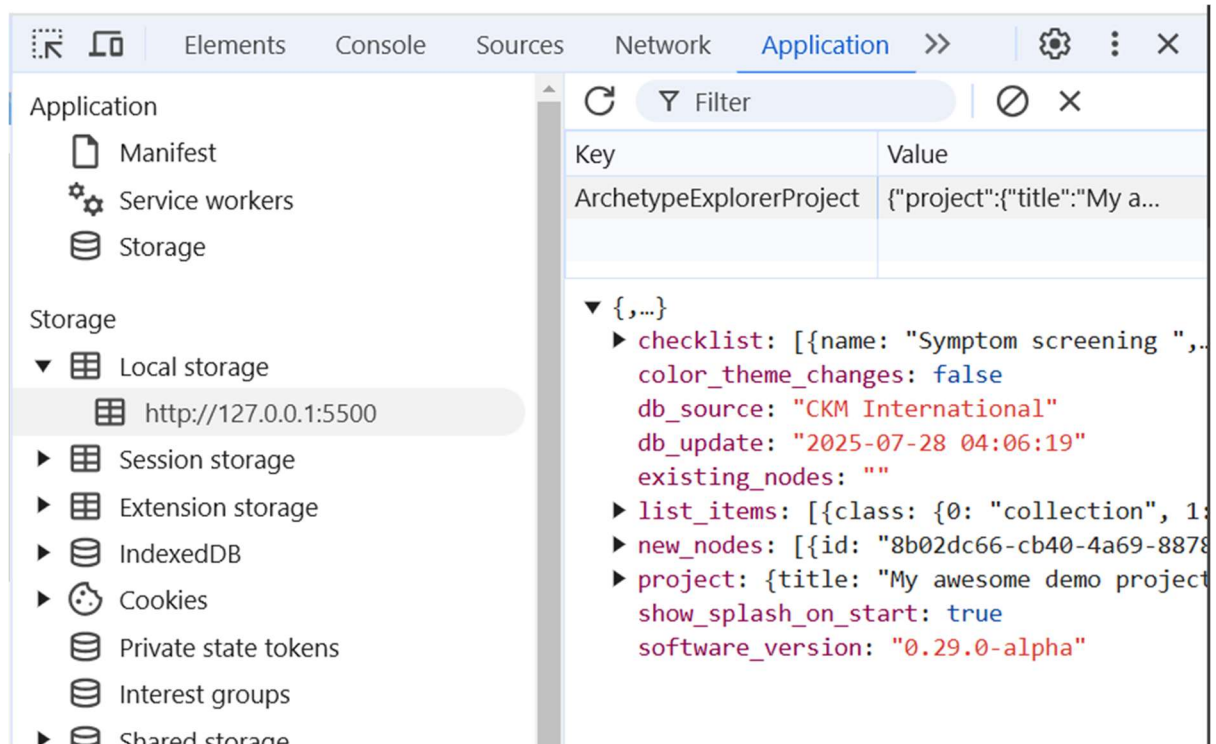
Local Storage

When using the visualisation tool, your project data is saved in your browser's **local storage** under the key `ArchetypeExplorerProject`. This ensures your work is not lost when you refresh or close the page*.

Here's how you can find, back up, or delete this data in different browsers:

Google Chrome

1. Open Developer Tools: Press `Ctrl + Shift + I` (Windows/Linux) or `Cmd + Option + I` (Mac), or right-click the page and select **Inspect**.
2. Go to the **Application** tab.
3. In the left sidebar, expand **Local Storage** and select your website's URL.
4. Find `ArchetypeExplorerProject` in the list. You can view, edit, or delete it by right-clicking the entry.



Mozilla Firefox

1. Open Developer Tools: Press `F12` or `Ctrl + Shift + I` (Windows/Linux) or `Cmd + Option + I` (Mac), or right-click and select **Inspect**.
2. Click the **Storage** tab.
3. Expand **Local Storage** and select your website.
4. Locate `ArchetypeExplorerProject` in the list. You can view, edit, or delete it as needed.

Microsoft Edge

1. Open Developer Tools: Press `F12` or `Ctrl + Shift + I` (Windows/Linux) or `Cmd + Option + I` (Mac), or right-click and select **Inspect**.
2. Go to the **Application** tab.
3. Expand **Local Storage** and select your website.
4. Find `ArchetypeExplorerProject` in the list. Right-click to edit or delete.

Safari

1. Enable the Develop menu: Go to **Safari > Preferences > Advanced** and check **Show Develop menu in menu bar**.
2. Open Web Inspector: Press `Cmd + Option + I` or select **Develop > Show Web Inspector**.
3. Click the **Storage** tab.
4. Expand **Local Storage** and select your website.
5. Find `ArchetypeExplorerProject` in the list. You can view, edit, or delete it.

***Note:** Clearing your browser's cache will remove this data.

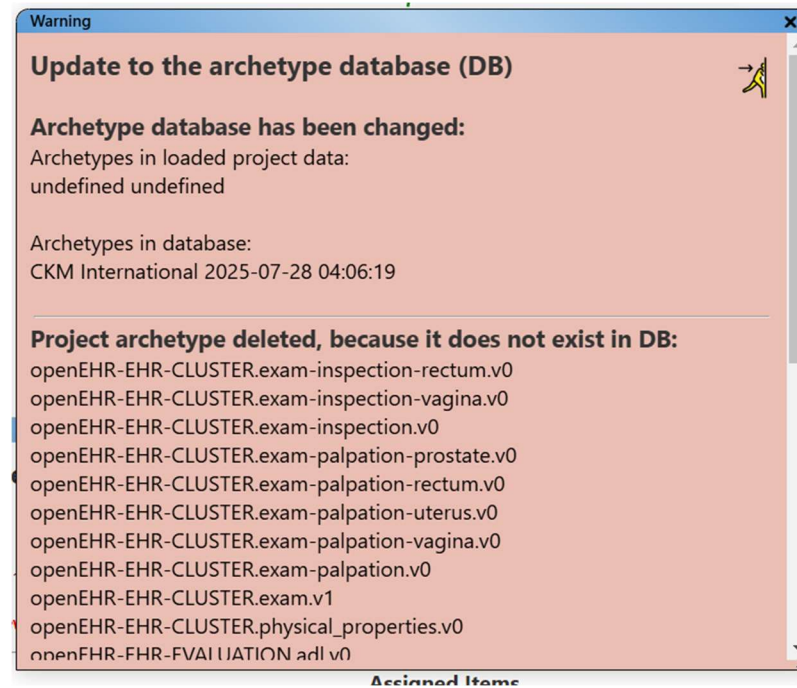
Archetype Database

The archetype data used in the online version of the Archetype Companion has been extracted from an archetype repository with the ADL Extraction Tool, developed for this project. Please refer for more information to: [ADL Extraction Tool](#).

Note: The information about openEHR archetypes presented in this tool has been extracted automatically from external sources (such as CKM) and is provided for convenience only. We do not guarantee the accuracy, completeness, or correctness of the extracted data, nor do we claim that it is up to date. This tool does not assert or imply any copyright ownership over the archetypes themselves. For authoritative information regarding authorship, copyright, and licensing, users are strongly advised to consult the original archetype source as stated within the tool.

The archetype data is stored in the "allNodes" variable, found in the "data.js" file.

Note: If the database is updated and there are changes in the archetype database, the database of the project is updated automatically. A detailed warning is issued in this case.



Update warning for the archetype database.

16.15. Running Archetype Companion Locally

1. Download the Package

- Go to the project repository: https://github.com/martinkochdesign/openEHR_fellowship_2025_project/tree/main
- Click on **Code** and select **Download ZIP**.
- Unpack the ZIP file to your destination folder.

2. Start Archetype Companion

- Navigate to the extracted folder.
- Double-click on `index.html` to launch the Archetype Companion in your browser.

Updating the Archetype Data Dictionary

1. Generate the Data Dictionary

- Open the `dist` folder inside your extracted package.
- Double-click on the latest version of the executable, for example: `create_existing_JSON_from_ADL_V4_15.exe`.
- Select the source for your archetypes when prompted.
- Click on **Download and create dataset.js**.
- The tool will:
 - Download a ZIP file from the selected source.
 - Extract all ADL 1.4 archetypes.
 - Compile the relevant information into a file named `dataset.js`.
- The `dataset.js` file will be saved in the same directory as the executable.

2. Update the Application

- Copy the newly created `dataset.js` file to the directory containing `index.html`.
- Restart the Archetype Companion if it is already running.

17. Annex III – Reflective Journals

17.1. Reflective Journal 1

On the Fellowship

It's incredible that a lot of time has passed in the fellowship. I would like to express my gratitude to everyone involved—this is a fantastic initiative, and I'm excited to become part of the “big picture” of openEHR, to see what others are doing, and to present my own results.

The presentations in the second month have been especially interesting. I completely agree with Heather Leslie's vision that a lifelong Health Information System (HIS) should be standard, and that we can address the challenges of real-world data with appropriate strategy and open-source standards like openEHR. For me, a major takeaway was the importance of applications for all stakeholders—medical professionals, patients, and data modellers alike, as was presented in the Catalan and Slovenian implementation of openEHR.

The mention of Microsoft's Common User Interface (CUI) was interesting from a tooling perspective. Although the original page is no longer hosted, I found much of the information on the Internet Archive: [Microsoft Health CUI Design Guide](#).

On My Project

Speaking of tooling, this brings me to my project. At the start, I filled out the proforma document, which was challenging in itself because it required detailed planning, which is very difficult when starting out. In hindsight, completing the proforma helped me focus on the project's goals and provided a clear outline, so I am quite happy with the resulting document.

Just to put you in context: In a nutshell, my project aims to create a tool (in the form of a webpage) to help modellers search for and organize openEHR archetypes.

Basically, a modeller with a list of clinical data elements on hand should be able to find appropriate existing archetypes and plan out the use of existing archetypes, the need for change requests in archetypes, and the proposal of possible new archetypes.

The project is divided into two main parts:

Backend Automation:

First, a program will be developed that periodically downloads archetypes from an established repository (in this case, the international CKM). Each archetype will be parsed, and relevant information extracted into a proprietary JSON dictionary. This process will be fully automated and invisible to the user.

Web Application:

The second part is a web application that uses this dictionary to offer the following functionalities:

- Fast search of archetypes and display of basic information
- Visualization of dependencies (parent-child, exclusions, inclusions)

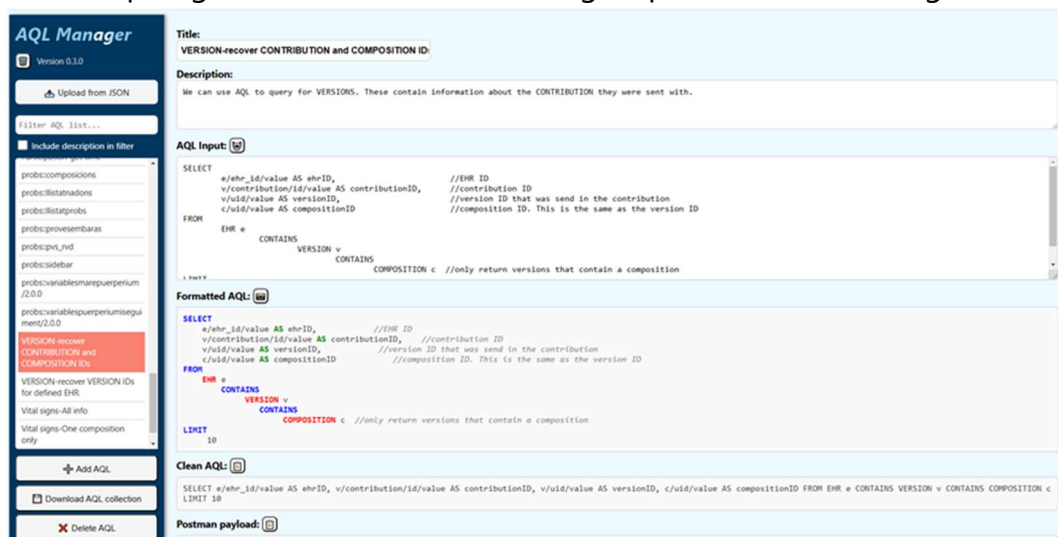
- Recommendation of similar or related archetypes
- Management of a collection of archetypes
- Creation of "new archetypes" as placeholders, so users know which archetypes are missing and need to be created
- Management of a checklist of predefined elements
- Project-based organization of all activities
- You can already see that this project will have many interconnected moving parts.
I'm also realizing that it will touch on various technical aspects (parsing, automation, web hosting) which will certainly expand my programming skills.

The development is based on my experience with two of my earlier projects:

The AQL Manager (Live online)

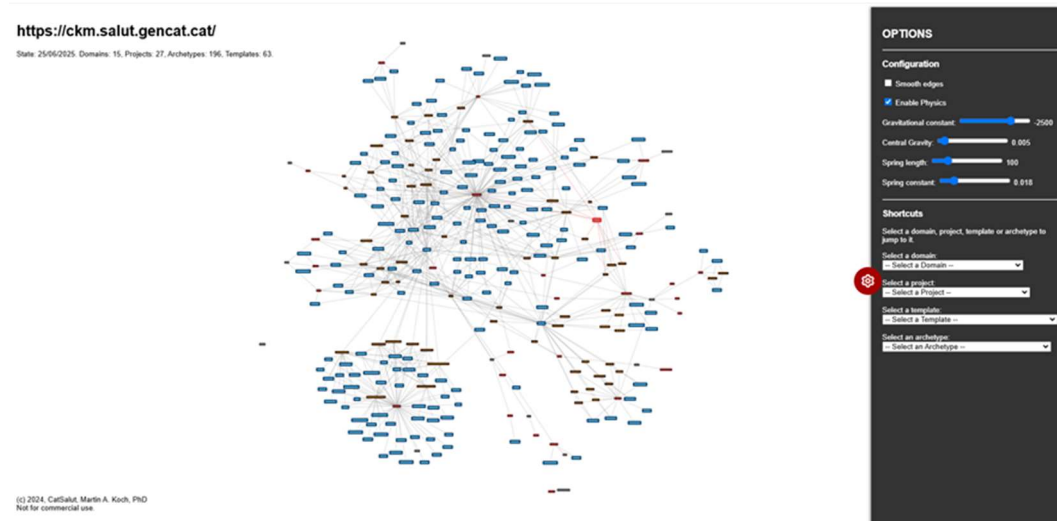
here: https://martinkochdesign.github.io/openEHR_AQL_manager/; Explanation

here: https://github.com/martinkochdesign/openEHR_AQL_manager)



Visualization of CKM contents (Example online)

here: https://martinkochdesign.github.io/CKM_content_visualization/)



Currently, I'm investing a lot of hours to reach a working prototype of both parts. The idea is to include as much user feedback as possible (if the feedback can be implemented in time is another story...).

I am always reminded of the aphorism of the "Ninety-ninety rule" (Tom Cargill, Bell Labs):

The first 90 percent of the code accounts for the first 90 percent of the development time. The remaining 10 percent of the code accounts for the other 90 percent of the development time.

This tongue-in-cheek observation leads me to speed development up in the beginning, because this might be beneficial to all the little details that a full-fledged GUI implicates in the end. However, this approach has an obvious drawback: it may create the impression that the project is "already done" early on, even though much work remains on the backend. If sometimes it looks like there are no changes in the tool, it is probable that I am working on some detail that is not perceived in the GUI.

As an interesting observation, I suspect that I am not only using the extracted archetype data for the tool, but I can also generate some valuable statistics, particularly regarding quality. Let me get the final numbers for the next journal or another publication, but I think I can pinpoint archetypes that might need minor changes to get a coherent format all over the International CKM repository.

Also, I've already noticed that about a third of archetypes lack keywords, which complicates semantic linking for the recommendation algorithm (don't worry, I am working on a solution!).

So, what are the planned next steps?

First, we need to finalize the proforma. I don't expect any major changes, as we have already discussed the project's scope.

Second, I have prepared an initial questionnaire, which I will use to gather feedback from

stakeholders. This will help us focus on the aspects that are most important to future users. As mentioned before, user feedback is crucial for this project.

Finally, I will begin compiling the documentation. Adding to the documentation as I develop will make it much easier to produce a comprehensive final version.

Thank you for following along with the project's progress. If you have any questions or suggestions, please feel free to reach out!

17.2. Reflective Journal 2

Month 6... Already?

Time has flown by since our last fellowship work session (and the reflective journal). I was well aware that time would be a critical factor in developing my fellowship project, but *whoa!* the pace has truly surprised me.



Wer hat an der Uhr gedreht? (*Who turned the clock?*)

The past months were shaped not only by the holiday season, but also by **EHRCON25**, which had a notable impact on my project work. On the positive side, I had the opportunity to demonstrate the current state of my prototype to Heather and Keisha. Our conversations sparked a wealth of ideas and additional content for the project.

On the other hand, though ultimately a good challenge, I was unexpectedly lucky enough to create and present a poster at the conference. This diverted some of my attention from the fellowship project itself, but the experience was rewarding. I particularly enjoyed this year's meeting, as I already knew many of the participants, especially fellow fellows and organizers.

Progress on the application has continued steadily. In fact, I have already published an overview video of the current version. You can watch it here: [Application Overview Video](#)

The next step is creating video tutorials, so the prospective users know every function of the application.

Month 5 fellowship session

I would like to extend my sincere thanks to all the presenters for the fellowship session on clinical modelling and community.

This month's presentations offered a comprehensive look at the intersection of value-based healthcare and clinical modelling. I found the explanation of VBHC especially clear, and I fully agree with the importance of focusing on patient outcomes and the role of openEHR models in enabling meaningful data capture.

The breakdown of archetype classes, referencing “An Ontology-based Model of Clinical Information,” was a highlight for me and will be useful in my own work. Heather's advice to resolve all data elements before starting modelling in openEHR tools resonated with my experience, as did the reminder to be mindful of copyright issues when reproducing information from archetypes, something I need to consider for my project.

Silje's presentation was a timely reminder not to lose sight of the fundamentals, even as I delve deeper into the technical aspects of templates and archetypes. Seeing the "DIPS form designer" in action was new for me, and I appreciated the practical perspective. Overall, these sessions reinforced the value of a structured, thoughtful approach to clinical modelling and the importance of community learning and sharing.

Continuing Documentation

Over the past month, I have continued to document the development process. While documentation can be time-consuming, I have found it essential to first define what the final product should include before diving into the actual work. However, there are times when it is beneficial to explore and identify potential challenges early on.

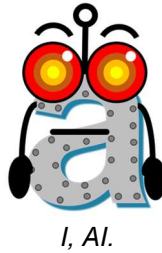
As a result, my current methodology is: **explore** → **document** → **develop**.

To make the documentation more engaging, I introduced a character to break up the text visually and provide the reader with occasional relief. I must admit, taking a break to create a simple illustration is not only enjoyable but also adds a touch of fun to the process.



AI, or Not AI: That Is the Question

As my fellowship project has progressed, one question keeps resurfacing: **Why not just use Artificial Intelligence (AI)?** It's 2025 - AI is everywhere, and it has proven itself capable of impressive semantic reasoning using Large Language Models (LLM).



Setting aside ethical and environmental considerations for a moment, the real question becomes: **What do we actually need AI to do?** And, just as importantly, what information must we provide to AI so it can deliver meaningful results?

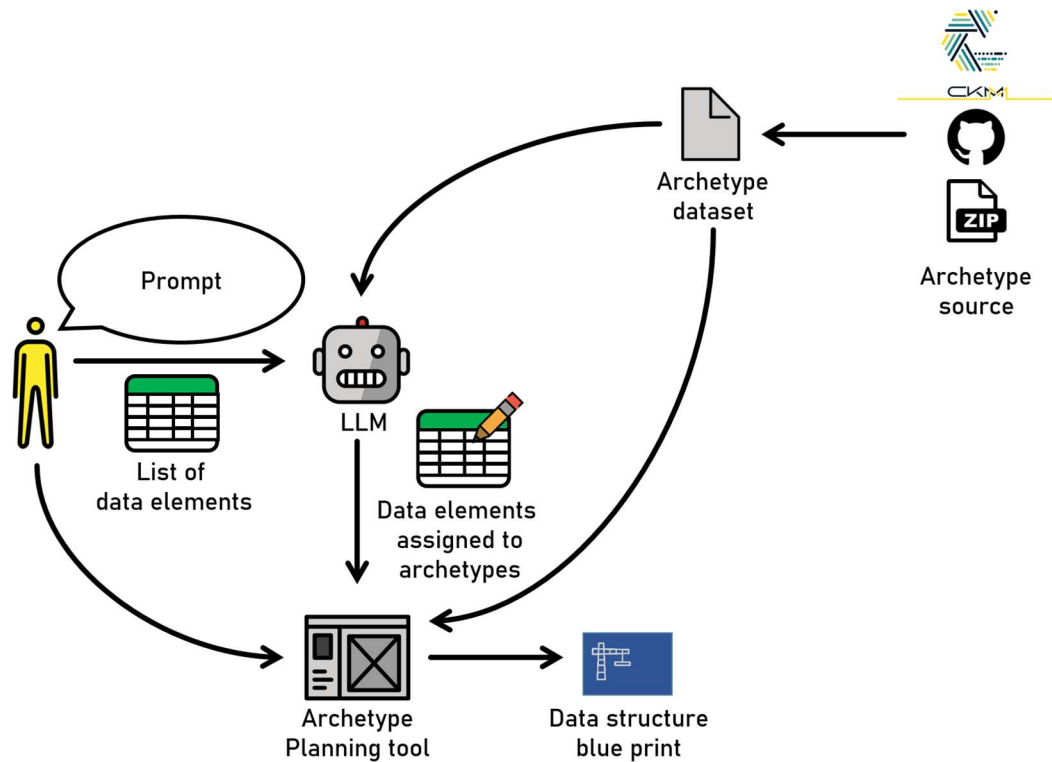
The original scope of this project was to develop a tool for searching, recommending similar archetypes, planning, and visualizing archetypes to form data structures from a given list of data elements. For most of these functionalities, using AI would be like “shooting sparrows with cannons” - the benefits would be marginal compared to more deterministic algorithms.

However, there is one area where AI could truly shine: **suggesting whole data structures by intelligently assigning data elements to archetypes from CKM or other sources.** This could give data modelers a valuable head start in creating those structures. The advantage of using AI here is its ability to consider all the semantic information embedded in archetypes and create meaningful links between data elements and archetypes.

For this to work, the AI would need more than just a list of data elements. It would require additional context to evaluate the semantic connections to specific archetypes. This could be achieved by providing a description of the intended application for the data structure, as well as detailed descriptions for each data element.

Here's how this could look in practice within the tool developed for this project:

- The user enters a list of data elements, supplying as much context as possible - both in the list and in the prompt for the AI (LLM).
- The AI analyzes the information from both the archetypes and the user's data elements, then suggests assignments of archetypes to each data element.
- This proposed data structure is then visualized in the tool, allowing the user to review and refine the AI's suggestions.
- Finally, the user can export the finalized plan and create openEHR templates in their editor of choice.



Using LLM (AI) in context of this project.

It's also important to recognize that not every data modeller will approach this process in the same way. Some may prefer to construct data structures from the ground up, valuing the deep understanding and oversight that comes from doing this manually. Others might appreciate having a proposed structure generated by AI, using it as a starting point to accelerate their workflow and refine the results. By offering both options, the tool could accommodate different working styles and preferences, ensuring that AI serves as a helpful assistant.

Whether this can be fully implemented during the course of this fellowship remains to be seen. Developing the data extraction and visualization tool as a one-person team - while also handling troubleshooting and documentation - is already a significant challenge.

Nevertheless, this overview outlines a possible path for AI integration.

Special thanks to Lars and Heather for inspiring me to explore the potential of AI in this context.

17.3. Reflective Journal 3

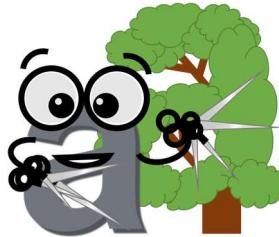
Continuing Work on the Tutorials

I have made further progress on the user manual, which is now available [here](#). In addition, I have created a comprehensive set of video tutorials to support the upcoming user test scheduled for December 2025. These tutorials can be accessed [here](#).

Working on the documentation has been a valuable exercise in distilling the essential information that users need. This process has enabled me to clarify the workflow within the Archetype Companion, ensuring that the steps are logical and easy to follow. By breaking down complex processes into clear, concise instructions, I have gained a deeper understanding of the user's perspective and the challenges they might face.

While scripting and recording the video tutorials, I encountered several UI/UX elements that were less intuitive than I had anticipated. This hands-on experience highlighted areas for improvement, and I took the opportunity to make immediate adjustments to the interface. These changes were aimed at enhancing usability and providing a smoother, more enjoyable user experience.

Overall, this phase of the project has reinforced the importance of iterative development and user-centred design. Creating both written and video documentation not only supports users but also serves as a valuable feedback loop for refining the application itself.



Customising towards the user's needs.

Presenting the Archetype Companion to the world

It is surprising how quickly time seems to be passing, even though, on the surface, it does not always feel that way. Looking back, I realise how much progress has been made, and yet there is still much to do.

At the beginning of November, I made the decision to initiate a user test with real users of the application. This marks a significant milestone in the project, as it moves from internal development and planning to actual engagement with the intended audience. According to my timeline, this testing phase will extend into the beginning of next year, allowing ample time for thorough feedback and analysis.

The planned timeline is as follows:

- **Mon 10/11/2025** - Mailing for volunteers
- **Mon 17/11/2025** - Reminder
- **Fri 21/11/2025** - Deadline for applications
- **Tue 25/11/2025** - Publish user test page
- **Wed 26/11/2025** - Contact accepted applicants
- **Mon 01/12/2025** - Official start of testing
- **Mon 08/12/2025** - Questionnaire accessible
- **Fri 19/12/2025** - Deadline for test & questionnaire
- **End of December-Start of January** - 🎅Holidays!🎅
- **07/01/2026**-Analysis of the feedback

I am hopeful that the feedback collected during this period will provide valuable insights. Ideally, I would like to receive suggestions for future development, as well as identify minor improvements that can be implemented before the project is concluded. Engaging with real users is both exciting and a little daunting, as it will be the first real test of the application's

usability and effectiveness in a practical context.

Looking ahead, I am aware that the next few weeks will be busy, balancing the ongoing management of the user test with other project responsibilities. However, I am optimistic that this process will not only validate the work done so far but also highlight areas for refinement and innovation. Ultimately, I hope this phase will contribute to delivering a more robust and user-centred application.

First user test concluded! The results are in!

By closing the user test in December, we can start the year with a clear view on the last month of the fellowship.

In general, the feedback was positive for the application.

Obviously, the application cannot be perfect, and the testing depended on the users clearly understanding the intended use of the application and any technical difficulties that lurk around every corner.

It stings a little when you receive criticism for something that you have worked on for so long, but in this case everything was very constructive.

The complete analysis of the feedback is documented here: [CLICK HERE!](#)

It has to be seen how far the identified problems can be addressed in the next month to give a better experience to future users.

Bigger, general changes to the intended use will have to be analysed, and the future development of the application has to be discussed further.

To all testers: a heartfelt thank you for the time, care, and honesty you invested in trying out the Archetype Companion. Your detailed feedback and thoughtful criticism are not only deeply appreciated, but will directly shape how the application evolves from here.

Fellowship session Month 7 - The Standards Landscape

I'm very happy to have participated in the session for month 7. I found both presentations extremely informative and useful.

The presentation by John Meredith (EY) was especially interesting regarding the pitfalls of interoperability and the benefits of an open system approach. Although the talk was quite rapid-fire and some things went over my head, the main ideas were clear to me. One key takeaway is that semantics are really important - interoperability always comes at a cost, because every data transformation destroys a part of the semantic meaning. The further we are in the development process, the more expensive it becomes to change the data pipeline. It's clear that everyone has to conform and standardize, but standards alone aren't enough. We need open platforms to actually operationalize open standards, and that's where FHIR and openEHR come in. The aim is to have a semantically coherent, canonical foundation, where the open nature of the platform enables a shared modeling effort.

The presentation by Luis Marco Ruiz (vitagroup) was very practical, which I appreciated. He

gave a detailed description of the data pathway, starting from patient consent on paper, moving to implementation in FHIR, including openEHR clinical data, and finally transmitting everything to a central repository for research. I found the discussion about the impact of German law on anonymization particularly interesting, as it showed how the legal framework dictates the technical solution-in this case, requiring double pseudonymization of the data.

Overall, I found both presentations valuable and thought-provoking.

Fellowship session Month 8 - Real World Cases

The Month 8 session with Luke Readman on One London showed me how hard it is to truly connect health and care systems, even within a single city. London is officially one region, but in practice it is a complex web of programmes, organisations and technologies that all need to work together. The London Care Record, the Universal Care Plan and the wider One London efforts illustrated how far reality is from the clean, integrated vision we often talk about.

The “Titanic strategy” warning really stayed with me: it is tempting to focus on the shiny, visible innovations above the surface while neglecting the large, invisible foundations underneath. Repeated reinvention and persistent inequities are, in a way, symptoms of this imbalance. For me, this connects directly to my openEHR work and clinical data repositories: advanced apps and analytics only make sense if they rest on a solid, resilient data and governance layer.

Leadership stood out as the key principle. Not as vague inspiration, but as the ability to define a clear shared strategy, insist on standards, keep equity in view, and resist being distracted by short-term wins. The One London Citizens' Summit also showed that leadership here includes engaging the public and building trust, not just managing technology.

This leads to a question I am still asking myself: how can we introduce innovation into such a complicated network without draining resources from building and maintaining the foundational infrastructure? In my own context, I feel a strong pull towards new tools and features, but this session reminded me that real progress depends on disciplined focus: investing first in interoperability, governance and robust data structures, and making sure that any innovation strengthens, rather than undermines, that base.

17.4. Reflective Journal 4

Month 10 session - Patients, Providers, Payers, and Policy Makers

During this month's session on *Patients, Providers, Payers, and Policy Makers*, I was particularly struck by how practical and outcome-oriented stakeholder analysis can be. The emphasis on understanding the value we are expected to deliver, and then engagement with users, decision-makers and policy makers, felt very relevant to openEHR work for the clinical setting. Ideas such as asking stakeholders "What data do you want to see?" and keeping the underlying data layer hidden, while focusing on the information and outcomes they care about were very interesting. Ensuring broad representation and good governance across all stakeholder groups also resonated with me.

The brief introduction to Zotero (<https://www.zotero.org/>) also gave me useful inspiration beyond formal research. Even though I am not currently doing extensive academic work, I can see clear value in using Zotero—or a similar tool—to organise documents, references and shared knowledge around openEHR, interoperability and digital strategy.

Month 11 session - Patients, Providers, Payers, and Policy Makers

This final fellowship session strengthened my understanding of *federation* and *research uses of health data*, especially using *openEHR*. Across the talks, I saw how *openEHR-based modelling, interoperability standards, and well-designed infrastructure* can enable practical applications such as *infection control, large research cohorts, and multi-site collaboration*.

Pascal Biermann's presentation on the *Smart Infection Control System (SmiCS)* from *HiGHmed* stood out to me, especially the way *openEHR templates* were turned into a usable tool for clinicians. The *network graph visualizations* of infection contacts and chains was right up my alley. I also learned about *SUS/cSUS usability scales*, which I hadn't known before, and realized this could have been useful for evaluating my own fellowship project.

Cecilia Mascia and *Giovanni Delussu's* talk on *BBMRI-ERIC* and the *colorectal cancer cohort* gave a detailed, realistic view of *research data harmonisation*. I enjoyed the clear walkthrough of the *conversion workflow* (from an existing database through transformations into *openEHR compositions*) and the challenges they highlighted, especially dealing with *variable input data quality*, which closely mirrored issues we face in my workgroup.

Reto Wettstein's overview of the *Data Sharing Framework (DSF)* in the *German Medical Informatics Initiative* was a highlight for understanding how *federated research* can work in practice. The technical explanation of the *peer-to-peer setup, security approach, and DSF node design* helped me see how *feasibility* and *data-sharing processes* can be coordinated across hospitals.