The open digital forensic platform Investigate. Innovate. Share.

The community vision states that Hansken is **the open digital forensic platform**, extensible by all users and in which parts can be replaced.

Hansken supports **investigations** (crime cases and intelligence) with focus on the efficient mobilization of all people involved (e.g., case investigators, digital experts, analysts, data scientists, advocacy, prosecutors and judges).

Hansken supports **innovation**, both research in the digital forensic field and continuous development of the platform itself.

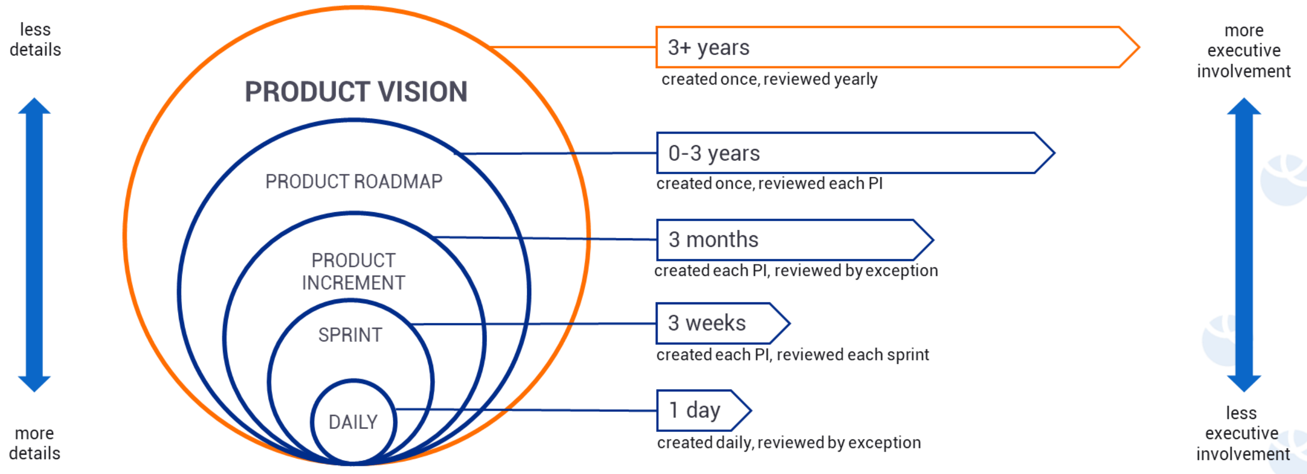
Hansken is a knowledge hub, to **share** knowledge on all levels, from methods and procedures, to tools and technology.

Hansken
The open digital forensic platform
investigate - innovate - share

Hansken product vision

The Hansken community asks for a broad and open digital forensic platform to support case investigations, from acquisition of data to reporting in court.

This product vision positions Hansken in this digital forensic community and directs future developments. It sets the frameworks and defines the **scope and architecture guidelines** for Hansken to support the platform design, implementation and decision making in the coming years.


less details
more details
PRODUCT VISION
PRODUCT ROADMAP
PRODUCT INCREMENT
SPRINT
DAILY
3+ years
created once, reviewed yearly
0-3 years
created once, reviewed each PI
3 months
created each PI, reviewed by exception
3 weeks
created each PI, reviewed each sprint
1 day
created daily, reviewed by exception
more executive involvement
less executive involvement

The most important requirements that force Hansken to be built amongst others on forensic principles, states that **Hansken can be used legally** and **the investigative results are accepted as evidence in court**.

Hansken operates within the legal boundaries.

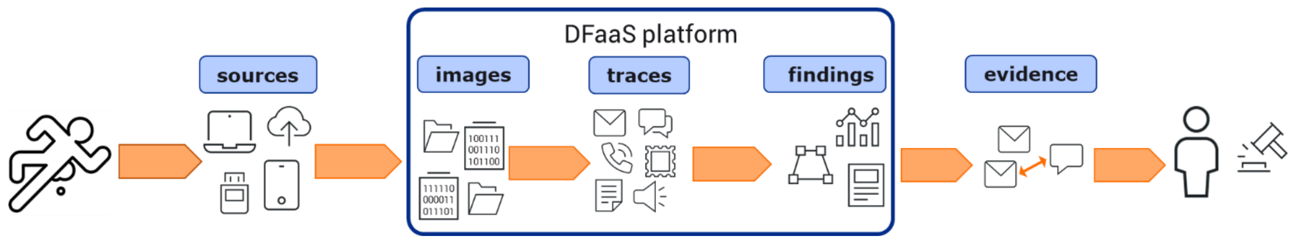
Hansken results are admissible in court.

Outline

1 Hansken's scope: Digital forensics

things we can store and offer via an API

Digital Forensics aims at accessing, analyzing and evaluating digital material in the context of criminal justice. To support this process with a platform, the domain is decomposed based on **things that can be stored and offered via an API** [1]: sources, images, traces, findings and evidence.



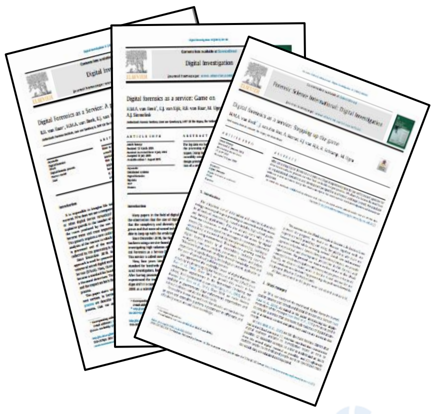
All of this, of course within the applicable legal boundaries and while maintaining the provenance of all results (i.e. keeping the forensic chains of evidence and custody intact).

Digital Forensics as a Service

Hansken implements Digital Forensics as a Service (DFaaS)[2], the fundament that defines the forensic principles (transparency). For legal purposes, it implements several privacy and security principles[3].

Three papers were published:

- **DFaaS: A game changer** [van Baar et al., 2014] presents the Hansken approach to offering digital evidence directly to case investigators and analysts and how this impacts the role of digital experts;
- **DFaaS: Game on** [van Beek et al., 2015] provides insight in the design principles, architecture and chosen technology for Hansken;
- **DFaaS: Stepping up the game** [van Beek et al., 2020] presents the lessons learned from offering DFaaS in the Netherlands for a decade.



2 Embracing the MACH design principles

Micro services
API first
Cloud native
Headless
machalliance.org

The MACH design principles[4] match Hansken's early design principles. They state:

M: (Microservices based) Hansken should be constructed from individual pieces of business functionality that preferably are independently developed, deployed, and managed;

A: (API-first) Hansken's functions should be available via well-defined application programming interfaces gateways;

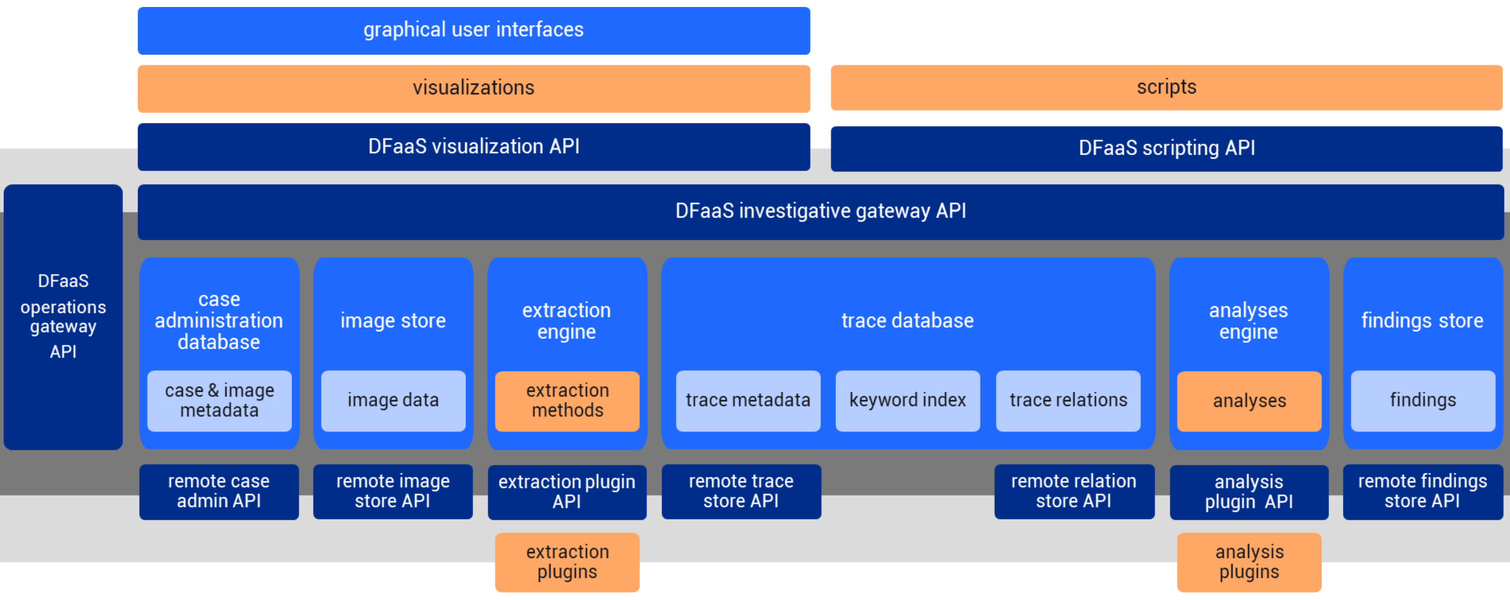
C: (Cloud-native SaaS) Hansken should build on cloud technology;

H: (Headless) Hansken's front-end user interfaces should be decoupled from its back-end logic.

A DFaaS platform like Hansken should include data stores, databases, processing engines, API's, visualization templates and central gateways:

- **Data stores and databases**[5] store and provide access to case administration, images, traces and findings
- Processing **engines** service the extraction of traces as well as their analyses;
- APIs and **visualization templates** form the base for reporting investigative and analyses results;
- Central **gateways** provides transparent access to all components of the platform.

[1] based on *A forensic visual aid: traces versus knowledge*, https://doi.org/10.1016/j.scijus.2018.08.006.
[2] The DFaaS papers are available at https://www.hansken.nl/publications-documents-and-links.
[3] The Hansken privacy and security measures are available in the online Hansken guide (via the Expert UI).
[4] More information on MACH can be found at https://machalliance.org/.
[5] Data stores typically contain raw material that can be stored and retrieved. Databases store structured data and provide complex functions to combine and/or retrieve the stored data.



3 The Open Digital Forensics as a Service Platform

In a forensic investigation, sources are identified (e.g., smartphones, laptops, wiretaps, cloud storage). Source recognition and preservation is not in scope for Hansken. Results from as DFaaS platform can support this step, though. Their data is partially or fully acquired and stored as forensic image data (e.g., EnCase images, UFED reports) and administered in cases.

Digital artifacts called traces (e.g. emails, pictures, documents, system logs, call details) are extracted from the images, following a strict trace model. Apart from the trace metadata describing the trace, a full keyword index is constructed of any piece of text that is recognized (including the metadata). Next to identifying trace properties (e.g., file name, email subject, document authors), traces can also be classified (e.g., music versus speech, objects in pictures), data can be enriched (e.g., by identifying text in pictures) and trace relations can be made (e.g., cyphertext/key, email/account).

These traces can be investigated (e.g., emails sent on a date, documents containing search terms, pictures with weapons). Also, general analyses on the traces (e.g., clock validation, desktop reconstruction, communication networks, timelines, communication networks) as well as case-specific ones (e.g., crime-related phrases in an email, geo-information at the crime scene) are captured in findings.

Finally, formal reports are added as evidence to case files. All these steps must adhere to the forensic principles and operate within the case-specific or cross-case legal boundaries. Since all organizations have their own way of handling and storing case files, the actual creation and storing of the evidence is not in scope for Hansken.

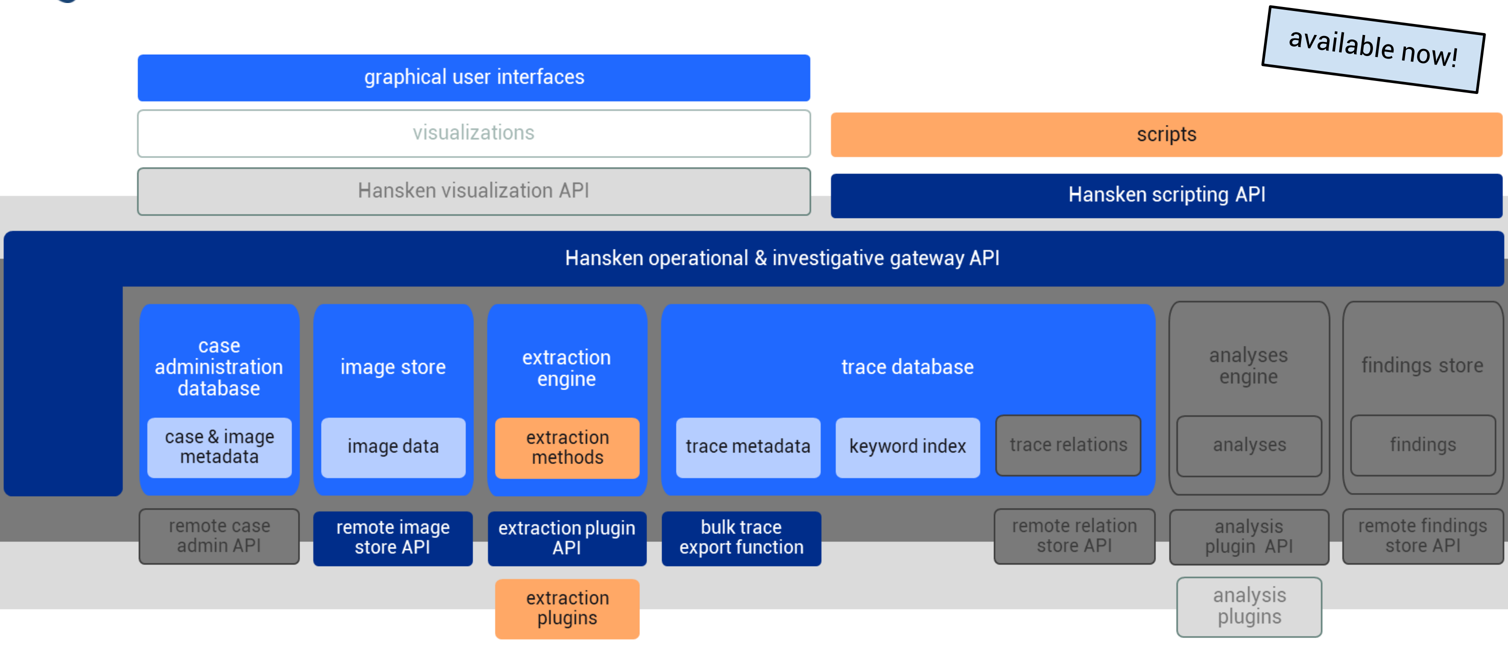
MACH dictates that a DFaaS platform like Hansken should implement all these business function in separate components (data stores, databases and processing engines) using cloud-native technology[6]. All data stores and databases are extended with an API to remotely access or store the data. The processing engines get an API to plug-in external tools/functions.

The complete the platform, it should support all these investigate actions via an investigative gateway API. Operators can manage the platform via an operational gateway API (preferably in a separate security domain so that the operational functions are not unnecessarily exposed).

The platform APIs are wrapped with a visualization API to easily and quickly develop visualizations contained in graphical user interfaces. Also a scripting API is provided for automating functions in scripts.

Hansken should build on cloud-native technology.

[6] An overview of the cloud-native landscape is available at https://landscape.cncf.io/.



4 Current Hansken implementation

Hansken implements several DFaaS services, following a modular setup, where functions are available via APIs and user interfaces are decoupled from the Hansken core platform:

Case administration database (Hansken project service) that provides flexible case administration functions to register cases and image metadata, both with unlimited properties. Currently, this service also stores case-specific Hansken logs, like extraction logs.

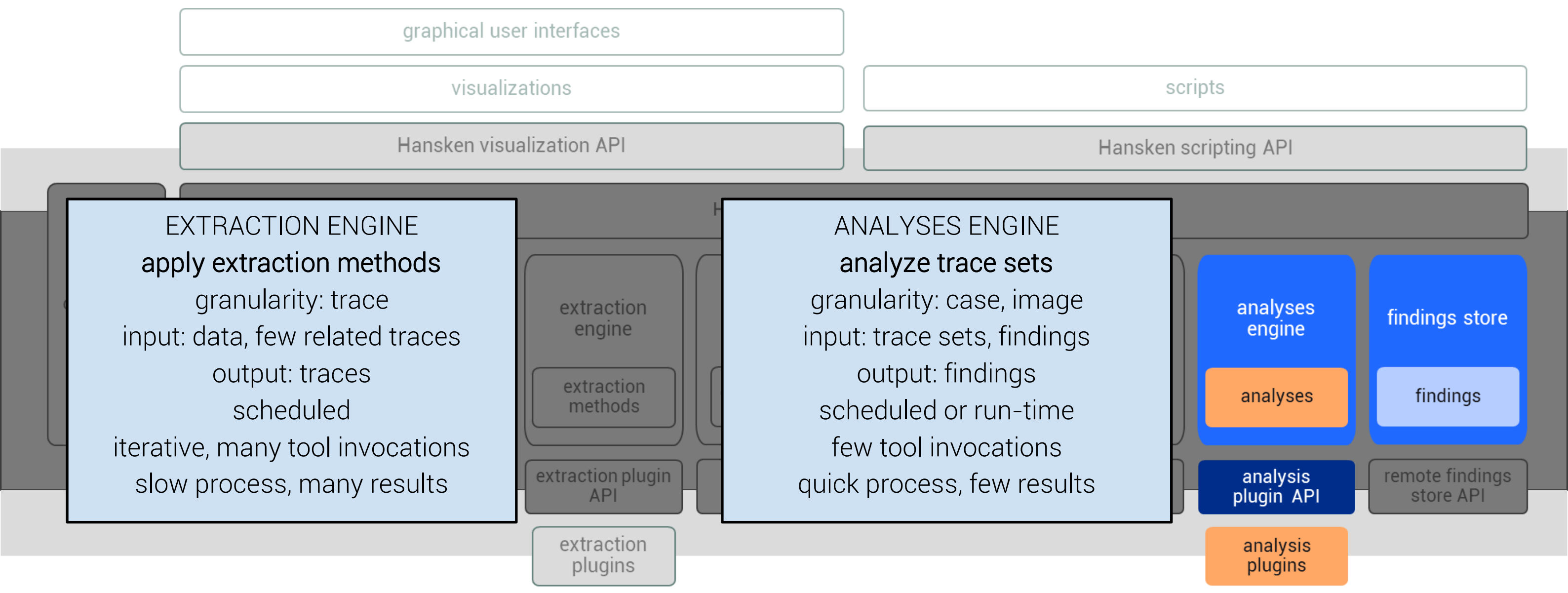
Image store (Hansken data services) for storing and accessing image data. Supported underlying storage solutions are HDFS, CEPH or a local file system. Next to the actual image store, (enriched) data from extracted trace can also be stored separately in an object store. To prevent data duplication, it is also possible to configure Hansken to directly use (secondary) remote storage for accessing image data.

Extraction engine (Hansken extraction service) for iteratively applying extraction methods (tools) to data (images) in order to give insight in that data. Hansken builds on Hadoop MapReduce, a third-party distributed processing solution. The extraction engine supports extraction plugins that can be offered in Docker containers. Traces can be exported during the extraction process using Hansken's bulk trace export function.

Trace database (Hansken trace service) for storing and retrieving indexed traces (trace metadata and a keyword index). Hansken builds on Elasticsearch, a third-party search engine technology. Trace data is described as a transformation on the original image (data virtualization) rather than copying them out and storing them separately.

Hansken operational & investigative gateway API (Hansken gatekeeper) serves all function available in Hansken as RESTful API. Routing within the platform is handled separately by the Hansken lobby service. This API provides both operational and investigative functions.

Currently, three graphical user interfaces are available on top of Hansken. Also, a scripting API is wrapping the RESTful API to simply writing Python scripts.



5 DFaaS platform: Support analyses and findings

Hansken should implement an analyses engine including an analysis plug-in API. To better support investigations, parts of the external analyses should take place behind the gateway API.

Hansken currently has no analyses engine. All analyses take place outside Hansken, using the RESTful API wrapped with the Hansken Python API or directly coded in the Hansken user interfaces.

An analyses engine enables analyses of the extracted resources: all traces including their (enriched) data. Such analyses are typically invoked once or a few times per case and typically take much shorter than extractions. The results of such analyses are called findings. Also, analyses might result in new traces, addition properties on existing traces and/or relations on the traces (see Section 6). Since some analyses require other analyses to be applied first (e.g., finding key players in a network requires a communication network), multiple run-levels can be defined.

Like for extraction methods (e.g., for identifying object in pictures), artificial intelligence can be used for analyzing trace sets. To maintain the provenance of findings, this puts high demand on registering the applied techniques and documenting the analyses.

Scheduled versus run-time analyses

Depending on their type, analyses can be scheduled, after which the results are stored (e.g., extraction reports stored as findings or communication network analyses stored as trace relations). Automation of these analyses must take place within the legal boundaries of the actual case.

Alternatively, analyses can be executed on run-time, after which the findings are returned to the (waiting) user and optionally stored (e.g. an overview of attached devices or the origin of an email). Several technologies exist to support such analyses (e.g., scripts or documents with live code).

Generic versus case-specific analyses

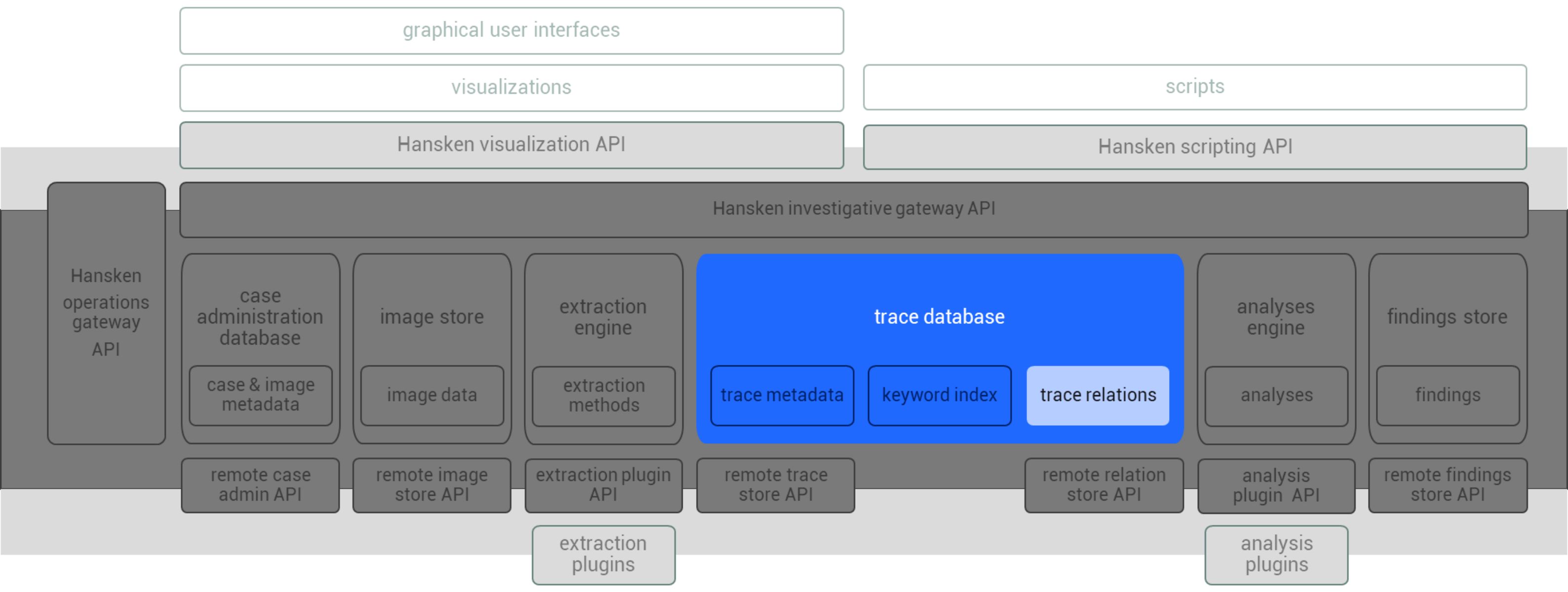
Generic analyses (e.g., clock validation) are developed in forensic software libraries with high quality demands on performance, testability (including reference data) and documentation.

Case-specific analyses (e.g., for identifying the origin of a specific email) take part in the investigative process. Such analyses are scripted and can be plugged-in into the analyses engine. Their quality can be assured by individual (external) reports made up by the responsible digital experts.

Storing and retrieving findings

Hansken should implement a findings store.

A findings store can be used for storing and retrieving findings. Findings can be stored as structured data (e.g., a table with statistical extraction information or a desktop reconstruction) or as a human-readable report (e.g., a clock validation report or a forensic report for a trace).



6 DFaaS platform: Support trace relations

Hansken should extend the trace database with functions to store and query trace relations.

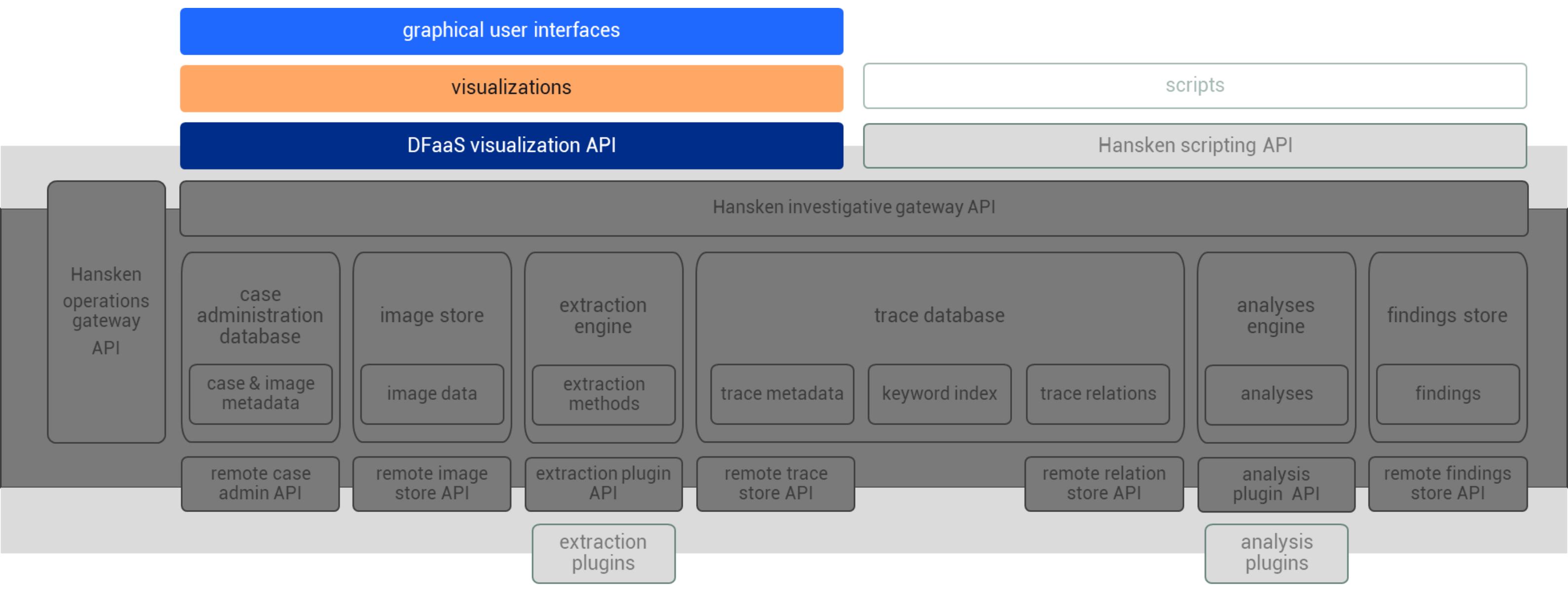
Traces are digital artifacts that support tactical investigations (e.g., emails, chat messages, office documents, pictures, step counts, geo locations) as well as technical investigations (e.g., file systems, connections, system logs, unallocated space).

Traces typically have multiple types (e.g., document/file, picture/attachment/geo-location) with corresponding properties (e.g., file name, picture camera, email sender). They can also have multiple data streams,, viz. typically, the actual raw data of the trace (e.g., the bytes assigned to a file), but also enriched data (e.g., the text extracted from a picture or PDF).

Single-typed traces with high occurrence are called tracelets (e.g., entities). Next to all properties (called metadata), traces can have data (e.g., the actual picture, body of an email or contents PDF document) that is made searchable using keywords.

Traces can relate to other traces (e.g., email networks, cypher text/decryption key). Such relations can have properties too (e.g. a relation between an email and an account can have properties stating that the account in the actual sender of the email).

These trace relations can result from both extractions method and (scheduled) analyses. They are key input for many successive analyses. Hansken does not yet support storing such relations and there is no function to search and filter them. Since relations are structured data possibly containing their own properties, they should best be stored in the trace database.



7 DFaaS platform: Modular and extensible visualizations

A customizable graphical user-interface should be available for Hansken.

Currently, three graphical user-interfaces are available on top of Hansken. None of them is customizable by users. To use Hansken, one or more graphical user-interfaces should be available. These interface should be user-friendly and cover as much of the functions in the platform as possible.

The user interfaces themselves should be as flexible as possible, such that users can construct their own dashboards. These dashboards can be either case-specific (based on tactical case information) or crime-specific (e.g., child sexual abuse cases, hacking cases, financial fraud cases).

Visualizations based on (automated) dashboards must take place within the legal boundaries of the actual case.

A visualization API should be available for Hansken.

Hansken provides access to all its function via its RESTful investigative and operational API gateway. A scripting API[7] is wrapping the RESTful API to simply writing Python scripts. Similarly, a JavaScript module[8] should wrap the RESTful API to simplify the construction of UI components.

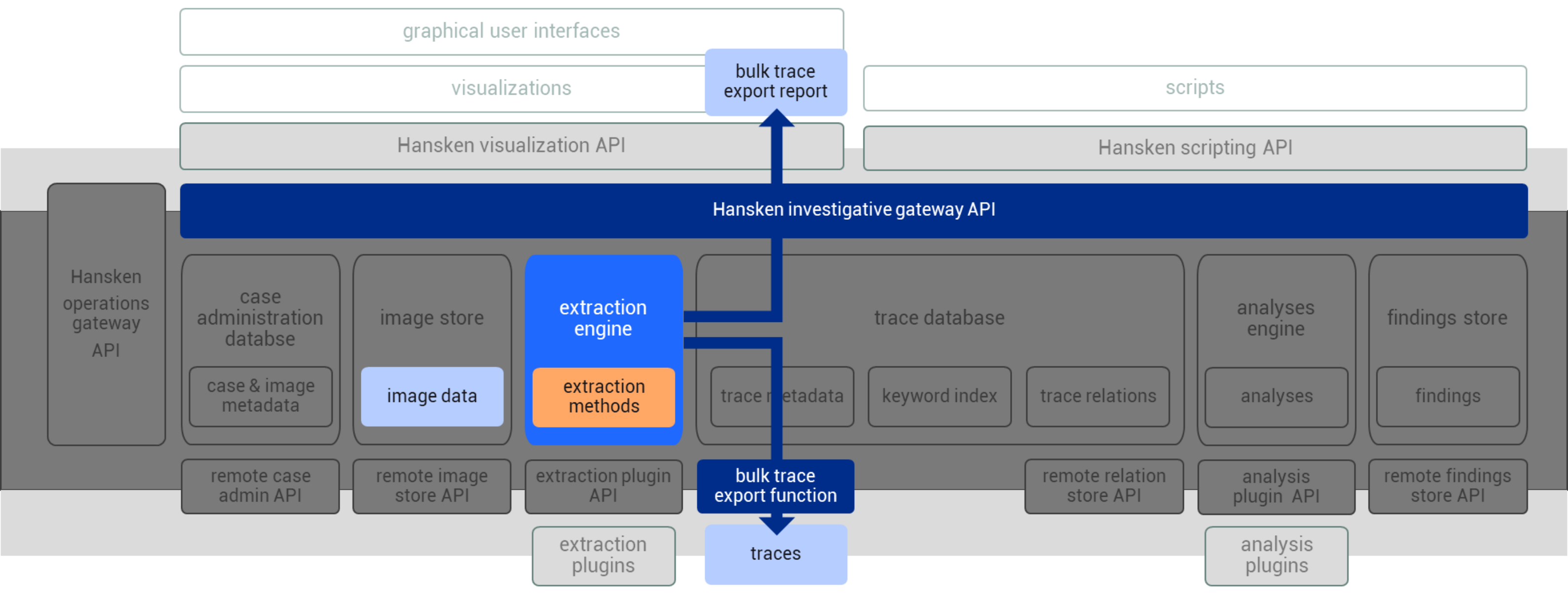
Reusable user-interface components should be available.

A framework with visualization templates (e.g., graphs, tables, networks, maps) should be available that enables users to easily define visualizations and adopt the interface to their demands.

This framework should build on the visualization API. In this way, web developers can quickly develop new features without the need for a detailed understanding of the Hansken RESTful API.

The Hansken Community should provide a central repository for sharing visualization templates. Each analysis should come with solid provenance and documentation.

[7] Hansken.py, available at https://pypi.org/project/hansken/.
[8] Hansken.js, available at https://www.npmjs.com/package/hansken-js.



8 Additional function: Bulk trace export function

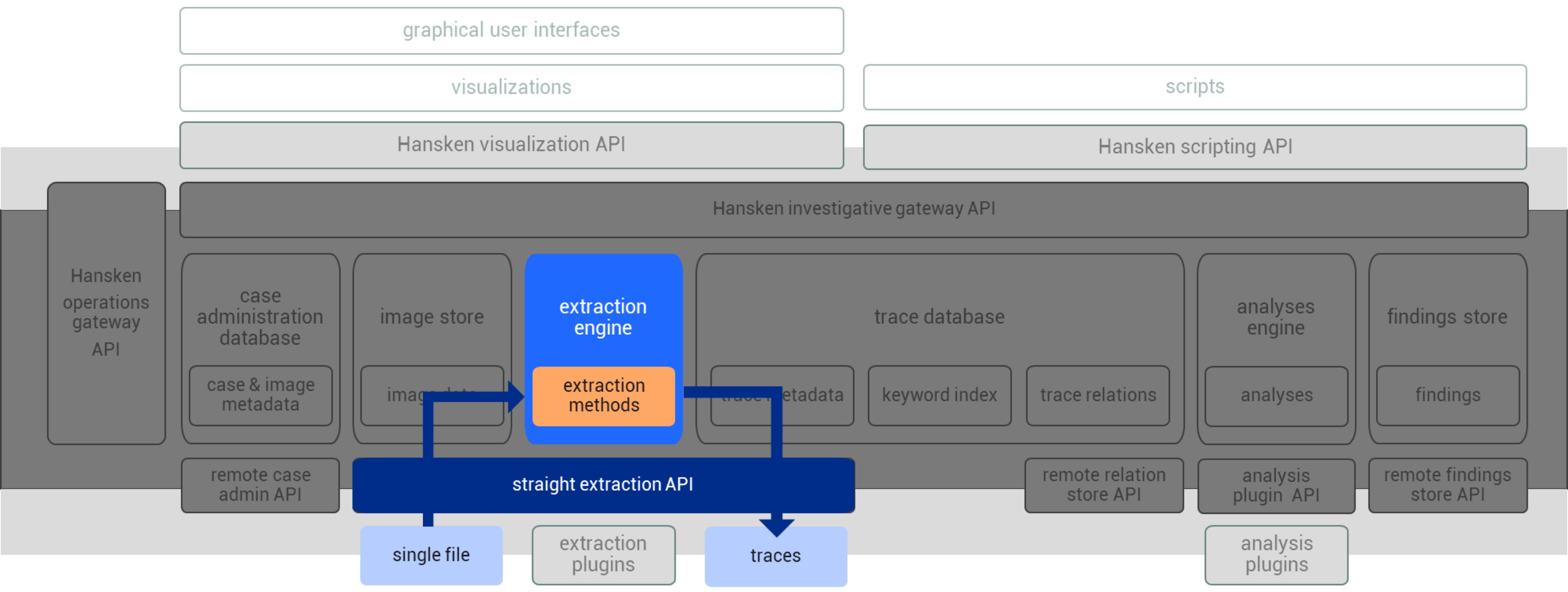
Extracted traces should be available for external processing during the extraction process.

Next to supporting case investigations, Hansken functions (especially the extraction service) can serve other goals. To process traces in their own application landscape, Hansken users require access to extracted traces as soon as possible. This lead to the implementation of a new Hansken function, the so-called bulk trace export function[9].

Authorized Hansken operators can configure and enable the export as part of the Hansken extraction. During the extraction, a dedicated tool in the extraction engine put traces directly on a remote queue.

Hansken is frequently challenged in court regarding issues such as security, privacy and transparency. This always involved the legal process (from seizure to court) and the role Hansken plays to support this process in a forensically sound manner. This export function impacts this process and can thus lead to additional discussions in court. Therefore, and since Hansken is a digital forensic platform, the export function comes with precautions to ensure forensic integrity of the results.

[9] The current Hansken bulk trace export function is a first implementation to put traces on a remote queue during the extraction process. It does *not* support all extraction functions available in Hansken. Results from deferred tools (tools that combine multiple traces by accessing the trace database in the end of the extraction process) and meta-extraction plugins (extraction plugins that only process trace metadata) are currently *not* available via this export function.



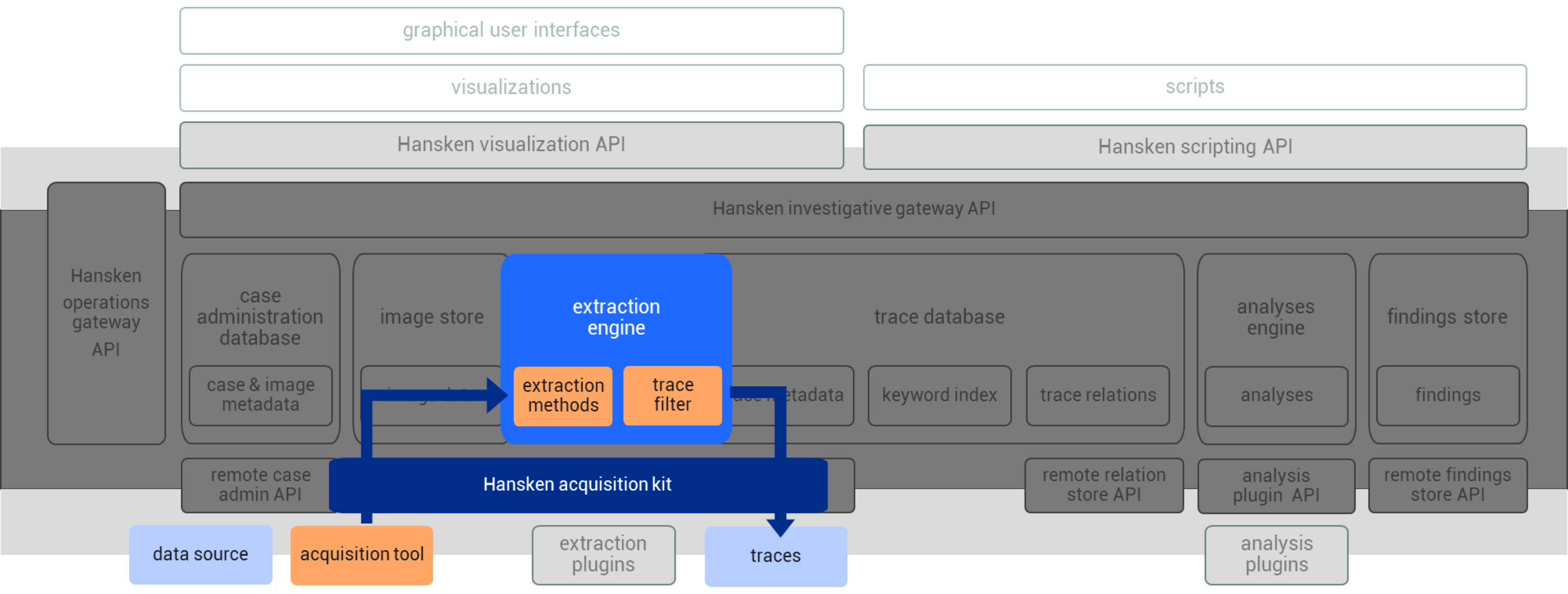
9 Additional function: Straight extraction API

Extraction engine capabilities should be directly accessible.

Currently, following the DFaaS process, after being scheduled via the operational gateway API, the extraction engine processes data stored in the image store, resulting in traces stored in the trace database.

The (embedded or plugged in) extraction methods in the extraction engine should be available via an API, such that an end user (typically a digital expert) can apply them to data outside the platform.

This makes it possible to use the (centrally maintained) forensic methods in any investigation, where the outcome (i.e., the traces) includes the chain of custody (i.e., the versioned methods that were applied).



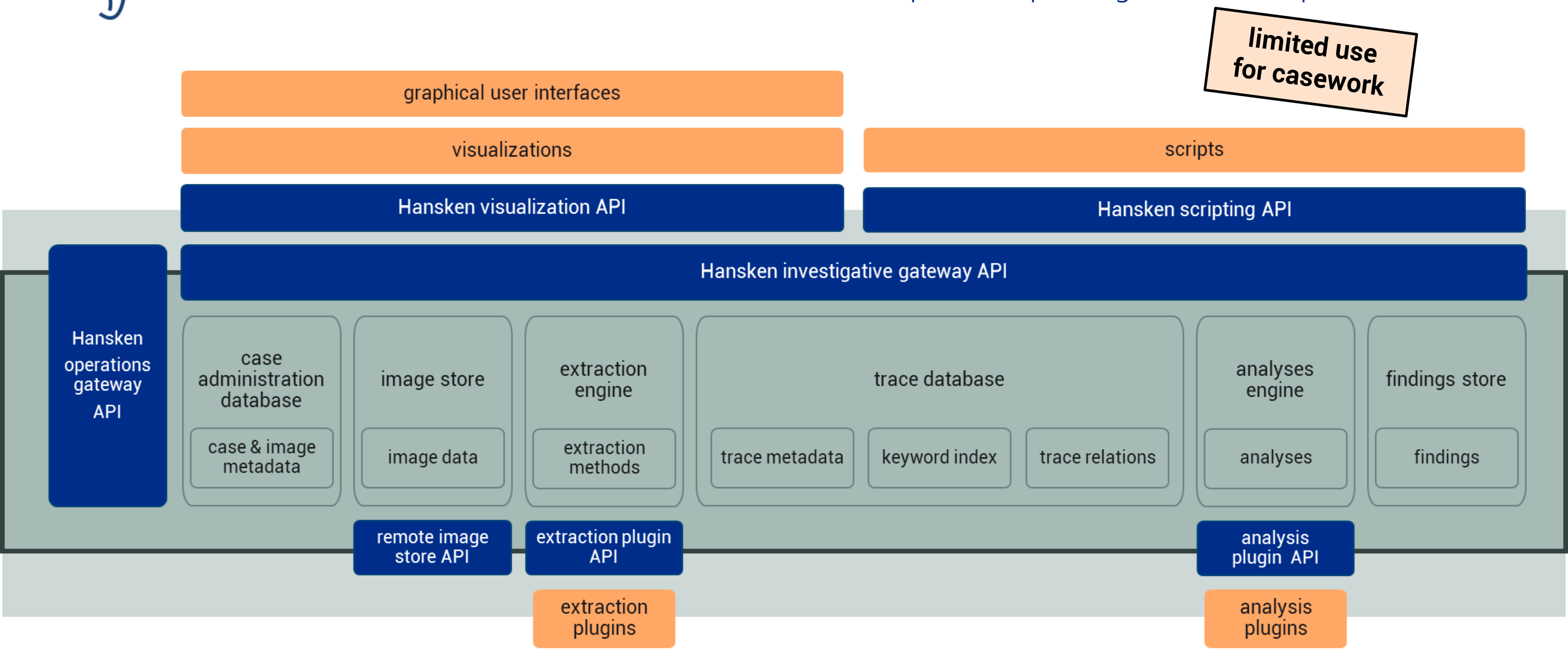
10 Additional function: Hansken acquisition kit

The Hansken extraction functions should be accessible for creating sparse forensic images.

The Hansken extraction engines provides automated extraction of traces from forensic images. Due to legal or judicial requirements, not all of the identified traces can take part in an investigation (amongst others, but not limited to privileged communication).

The restriction should preferably be implemented during the acquisition phase (where Hansken currently does not take a role). The filtering can be based on white listing (only traces that match a filter should be acquired) or black listing (traces that match a filter should *not* be acquired).

A Hansken Acquisition Kit can process data, extract traces, apply the filtering and store the results in a (logical) forensic image.



11 Single machine Hansken Development Kit

Simplify the development of extensions to Hansken to the greatest possible extent.

Any digital expert, software developer, web developer or data scientists but also case investigators and analysts can build their own extension to the Hansken platform (extraction plugins, analyses plugins, scripts, visualizations and graphical user interfaces). For this, Hansken provides plenty of APIs.

The Hansken APIs should be self-contained, versioned and separately released.

To simplify development of components on top of these APIs, Hansken and its documented APIs should be available in an environment that is under control of the developer to the greatest possible extent. A development kit that can run on the local machine of the developer maximizes this control. In this environment, developers can process their own data and develop, test, monitor their Hansken extensions prior to using them in production.

The development kit open functions that do not adhere to all legal demands while investigation case data (e.g., auditing, filtering privileged communication), making this available for limited use for casework.

May 2022.

This product vision is the result of many discussions and sessions, amongst others with staff of the Netherlands Police, the Dutch Fiscal Information and Investigation Service (FIOD), the Dutch National Prosecution Service, the Dutch Ministry of Interior and Kingdom Relations, the Netherlands Forensics Institute, the Hansken development teams, the Hansken architecture group, the Hansken product manager and the international digital forensic academic community.