ESSnet Smart Surveys

Grant Agreement Number: 899365 - 2019-DE-SmartStat

Link to our CROS website

Workpackage 2

Smart Survey Pilots

Deliverable 2.2: Modularity from the viewpoint of consumption (WP2.2b)

Version 1.0, 23-03-2022

Prepared by:

Tom Oerlemans, Barry Schouten (CBS, Netherlands)

ESSnet co-ordinator:

Shari Stehrenberg and Markus Zwick (DESTATIS, Germany)

smartsurveys@destatis.de

Workpackage Leader:

Barry Schouten (CBS, Netherlands)

Jg.schouten@cbs.nl

telephone : +31 70 3374905

SUMMARY: This deliverable discusses modularity of the Household Budget Survey app evaluated within the case study of work package 2.1 (Consumption). While designed for the Household Budget Survey, a mandatory survey in the European Statistical System (ESS), several components are configurable and may serve other data collections. Such modularity is evaluated from three viewpoints: a) the application to surveys with a similar set of specifications, b) the application to surveys with a broader perspective, and c) the application to surveys with a more focussed perspective. The boundaries of modularity are discussed in each setting. The most important boundary is that of the underlying machine learning procedures used to read, interpret and classify receipts. Some directions are given on how to further improve modularity. The reader is referred to deliverables 2.8a and 2.9 for detailed descriptions of the app frontend, backend and methodology.

1. INTRODUCTION

Modularity is an important feature of data collection tools. This holds especially true for tools that assist smart surveys. Smart surveys include smart features that often, if not always, require advanced processing of new forms of data and imply complicated trade-offs in respondent interaction and quality control. Development of such methods, often taken from the toolset of machine learning methods, requires a considerable investment in time and resources. It is both effective and efficient to be able to employ such smart features and accompanying methods across different surveys with similar themes and output needs.
The current HBS app has two smart features for which modularity is important: advanced product search routines and receipt processing. For product search, it applies string matching between respondent-typed answers and products lists. For receipt processing, it concerns machine learning methods to extract text through optical character recognition (OCR), to interpret texts through language processing, and to classify extracted products to formal categories such as COICOP. A third smart feature, uploading digital receipts is being explored and would add another challenge to modularity.

Obviously, modularity is not just about smart features. A survey has various design features that one would like to be configurable. In the context of consumption, examples of such features are the length of the reporting period, the type and granularity of purchases to be submitted, the type and number of intro questions and the focus on household or individual. Migration to another context, i.e. changes on these design features, should be relatively straightforward and fast. Within WP2 (see deliverable 2.7) four levels are distinguished: architecture (CSPA), methodology, logistics and legal-ethical. While modularity is often looked at from the architectural level, it must be evaluated and ensured from all levels. What must be seen as ‘straightforward’ and ‘fast’ is subjective, but the notions should apply to all levels simultaneously. In other words, moving from one setting to another should ideally imply only changing already configurable settings in the frontend and backend, only modest retraining/updating of machine learning methods, employing the same expertise of staff, working with existing data protection impact and risk assessments, and not having to submit a new ethical review board request. In evaluating modularity of the HBS app all four levels will be considered.

To understand the boundaries of modularity, three settings are explored. The first setting is where the HBS app is employed ‘as is’ with roughly the same output detail and granularity. An example could be a household budget survey as it is conducted in countries outside the ESS. The second setting is where

the HBS app is part of a broader survey requiring less detail and granularity. An example could be a health/life style survey in which consumption is just one determinant. The third setting is where the app is part of a more focussed survey demanding for more detail and granularity. A consumption/food survey is an example of this last setting where output needs may be detailed to actual consumption, metrics and distinctions between ecological and non-ecological products.
The first setting shows some overlap and similarity to shareability of a smart survey where the same survey is conducted by different institutions or in different countries. In deliverable 2.5 shareability is discussed for all WP2 pilots as a whole within the setting of the ESS.

The deliverable is organized along three sections. In section 2, the HBS app including configurable features is described. In section 3, the three modularity viewpoints are evaluated. Section 4 ends with a discussion and optional next steps to increase modularity. The reader is also referred to deliverable 2.2b in which modularity is evaluated in the area of time use.

2. HBS APP MAIN COMPONENTS

This section describes the Household Budget Survey app, version 2.1.16, as it is released in April 2022. Documentation can be found in deliverables 2.8a and 2.9 of ESSnet Smart Surveys and at

https://gitlab.com/tabi/projects/budget

The HBS description in this deliverable is limited to the configurable features. A distinction is made between frontend, backend, receipt processing and other services.

2.1 Frontend

The HBS app includes an intro questionnaire and a diary. Households are invited to report purchases within a specified reporting period. They can do so through manual data entry of individual products or through scanning of paper or digital receipts. A third option, uploading digital receipts is explored at the time of writing of this report. Respondents can edit purchases and they are presented with an in-app overview of the purchases they have reported aggregated through COICOP categories. Multiple household members can log in on the same credentials and submitted data is synchronized across devices. The app can be used on a smartphone and a tablet. Extension to other devices is being developed and coincides with the inclusion of multiple questionnaires on specific recurrent cost categories. The current app includes only one questionnaire.

Manual data entry employs two lists, a product list and a shop list. The product list contains common language product names and a unique link to the COICOP classification hierarchy. When respondents type product names, then through string matching a list of close matches is created and displayed. When a product is selected from one of the displayed matches, then also the COICOP is linked. When a product is not found, then the respondent can mark it as new and he/she is asked for a basic classification. The product is flagged as new and later in post-survey analysis set aside for expert classification.

Scan data entry has four steps. While photographing, the app searches for a receipt and overlays a bounding box. After photographing, the respondent can adjust the box and can retake a picture if needed. When the respondent accepts, then in-app OCR and language programming are performed and results shown to the respondent. The respondent can edit the results directly or later. As a last step, the respondent is asked to provide extra information on the receipt such as date, shop name/type and total amount. When editing products, the respondent can make use of the in-app product lists. The shop name/type is matched to the shop list and respondents can select from the displayed close matches or again mark the shop as new. Shop types are used for two purposes: receipt language programming and the in-app insights. Receipt formats strongly depend on the shop name/type. Since no complete real-time in-app classification is performed, the household insights make use of the shop type classification.

What is configurable?

- App name: The app name and institution logo can be modified
- Intro questionnaire: Content of the intro questionnaire, including the option to omit the questionnaire altogether, where question text and answers are loaded from a CSV file in a specified template.
- App UI language: A separate CSV file in a specified template is loaded identifying screen names and action labels per screen
- Product list: A CSV file in specified format is loaded
- Shop list: A CSV file in specified format is loaded
- Product search parameters: Various parameters can be specified in the Jaro-Winkler product search strategy.
- Paradata: Navigation behaviour logging can be enabled/disabled
- In-app editing: The option to allow respondents to edit product-price extraction in-app can be enabled/disabled
- In-app image cropping: Cropping of in-app images can be enabled/disabled
- In-app lower threshold to OCR performance: An OCR performance score is estimated in the app and a lower threshold can be set. Once the estimated score is lower than the threshold, the app prompts a message urging the respondent to retake the picture.
- In-house OCR language: A language or mix of languages can be selected to perform the in-house OCR.
- Insights: In-app insights can be turned off or delayed
- Reporting period: The reporting period can be set to any number of days/weeks

A config file is used to create the right settings. The config file is based on group names which on its turn consists of lists of user names. In the same fieldwork period user names can be assigned to different groups, leading to different settings. Hence, experimentation and randomization across different conditions can be prepared through the group names and config file.

2.2 Backend

The HBS app is designed from the perspective that services are replicated, i.e. it is assumed that an institution using the app creates its own backend and back-office. Docker containerization, as explained in deliverable 2.9, can be used to export/deploy the existing backend to another server and corresponding environment. However, links to other data collection systems need to be prepared by

the institution. For this reason, the more advanced components have only been developed and elaborated to a basic extent.
The back-office consists of a server and database linked to a case management system, i.e. the backend, a monitoring application, a receipt processing server, and an analysis server. The receipt processing server is described separately in the subsection 2.3. The analysis server is discussed in subsection 2.4.

The configurable features of the frontend are managed through the backend. The backend database consists of a fixed structure for which some views/tables are only relevant under certain parameter settings.

2.3 Receipt processing services

Automated receipt processing is currently the most important smart feature and deserves separate attention. It is closely linked to machine learning methods for text extraction and product classification.

The HBS app supports in-app preparatory OCR and product-price extraction, but it hinges on the assumption that receipt processing is done in-house. The main reasons are that 1) product classification in NL employs scanner transaction data that cannot be send to respondents, and 2) corresponding machine learning models would imply that the app grows substantially in size. The in-app OCR and language processing serve two main goals: 1) interaction with the respondent on minimal scan quality, and 2) interaction with the respondent on detected products and prices. The first goal is linked to the OCR performance score that needs to be larger than a pre-specified threshold. Below the threshold the respondent is asked to scan the receipt again. The second goal is based on respondent in-app editing of extracted candidate products and prices. Empirical results show that respondents indeed edit product-price extraction results and the edited data are of better quality than without. Since text extraction performance strongly impacts classification performance, in-app editing is preferred, although it adds some burden to respondents. The app thus provides the receipt scan plus products-prices possibly edited by respondents.

The in-app OCR and language processing have language dependencies. When the respondent selects a language, then this is used to inform OCR and to set the choice of keywords and monetary currency in language processing.

The in-house receipt processing is based on scripts in Python using a number of fairly standard libraries, see deliverable 2.9. A separate server launches queries to the HBS app database in order to detect whether new receipt data have been submitted. The receipt processing server, thus, is in the lead. This is done, because it allows for semi-automated processing where a data scientist is standby to check performance and progress. The in-house processing still is in a development phase when it comes to product dynamics and the need for online learning and retraining. The in-house processing has five main steps: 1) image pre-processing (enhancement of contrast, rotation if needed), 2) OCR, 3) search for start and end line based on keywords, 4) product-price derivations including discounts, and 5) classification. Steps 1 to 4 are only mildly countr/language-dependent. However, the classification step strongly depends on what data are available. The following options are possible:

1. Product descriptions (EAN/GTIN) linked to COICOP are available and machine learning training is done by a mix of string matching and manual annotation.
2. Receipt texts are manually annotated and coded to COICOP and serve as direct inputs to machine learning models.
3. Printed receipt texts are provided by stores including their EAN/GTIN code, are manually coded to COICOP and string matching is performed.
4. Printed receipt texts are provided by stores including their EAN/GTIN code, links to COICOP are available, and string matching is performed.

In NL, options 1, 2 and 4 are all possible and currently a hybrid approach is adopted. First, string matching is performed, but, when distances to words are too large, a trained machine learning model is applied.
The range of options varies per country. The most intensive is option 2, where a large number of receipts needs to be annotated to get acceptable machine learning performance.

An important feature of machine learning approaches is the need for retraining, either automated as a form of online learning or semi-automated as a form of active learning. For HBS, retraining is needed because product ranges in shops vary over time. Three events are distinguished: births, deaths and conversions. Births are products that are new on the market; they typically get assigned a new EAN/GTIN code. Deaths are products that are taken off the market; their EAN/GTIN is no longer in use. Conversions are products that keep the same EAN/GTIN, but change descriptions on receipts. Typically, conversions still resemble original names, but even so may require being added to train data. Products that vanish need to be removed form train data.
Retraining of machine learning approaches has a close link to modularity; once a different set of products is in scope, performance of models may go down and machine learning models may need to be revised.

Deliverable 2.3 of ESTAT-funded project @HBS2 gives a detailed account of the various options and provides a guide to corresponding code in Python. A link and a basic description can be found in deliverable 2.9 of ESSnet Smart Surveys.

2.4 Analysis services

An app-assisted HBS, in particular receipt scan data entry, offers new possibilities but also some challenges in analysis and processing of data. In deliverable 2.8a, a monitoring web application is described that has been developed for HBS. Furthermore, R and Python scripts are made available through the HBS git repository.

The monitoring tool and scripts are to some extent agnostic to the language and country as they are based on the HBS backend database. They do need modification when changing institutions or settings; they employ three background files that are context-specific:

- A sample CSV file that functions as backbone and allows for linkage to other files maintained by the institute carrying out the HBS.
- An admin data CSV file carrying sample ID's and a range of auxiliary variables available from administrative data or other sources
- A fieldwork results CSV file including paradata on the type of response or nonresponse,

The scripts make certain assumptions about the format and content of the files, e.g. there should be a variable 'insample' indicating whether households where identified as eligible. However, the admin data file likely changes form one institution to the other. The pre-processing of auxiliary variables from these data sources needs to be programmed anew each time the setting is changed.

The HBS app is configurable on in-app navigation logging and on offering in-app OCR editing by respondents. The scripts assume that in-app navigation logging is on and corresponding parts need to be turned off. In the monitoring application this is automated but it is not yet in additional Python and R scripts. Since definitions of in-app activity and diary completion currently depend on these paradata, at least some minimal in-app logging needs to be maintained. Turning of in-app editing by respondents has no consequence for scripts, but any derived data and statistics need to be handled as completely raw.

The analysis stage is also the stepping stone to retraining of machine learning models. Manual data entry, as explained above, allows for assistance through product lists linked to COICOP, substantially reducing burden in post-survey manual coding. When products are not recognized by respondents or not contained in lists, then respondents can add those and need to label them. The labels are merely temporary and the new products are separated afterwards for manual coding. For the scan data entry, retraining is needed as well when products are entered that have a large distance or low classification probability to any known product or COICOP category. Such products need to be added to train data, unless the OCR and language gave issues. Even with formal information from stores on receipt texts and or product descriptions, misclassifications may occur because local shops do modify texts or add local products. So, there are two types of retraining: 1) updating of machine learning models due to known dynamics in product ranges identified in available store product descriptions, and 2) extending machine learning models to unknown dynamics that come in through HBS data themselves.

3. MODULARITY

In this section, the three modularity scenarios are discussed. In each setting it is asked what may change and what are boundaries in each of the four levels architecture, methodology, logistics and legal-ethical. In this section, it is assumed that the survey is conducted in a country/language for which the HBS app has already been designed. When also the country/language is shifted then additional changes are needed. Those are described in deliverable 2.5 on shareability.

In the following, the term 'product' is used to refer to all types of products and services that a household may report.

3.1 As is in a HBS-type survey

What may change?

- The applied product classification
- Restriction to a subset of product categories
- The reporting period
- The (start) questionnaire content
- The recruitment and motivation modes, in particular the involvement of interviewers

- The type of incentives
- The look and name of the app (store name, logo, colour scheme)
- The requirement to delay household feedback on statistics
- The integration into a mixed-mode strategy, in particular the desire to allow respondents to submit receipts by postal mail
- The backend URL and services linked to it
- Type and form of monitoring of incoming data
- Absence of trained machine learning models for classification
- The requirement to perform semi-automated receipt processing
- Another helpdesk and support options
- The legal basis or legal constraints may be more strict, e.g. within household anonymization may be demanded or surplus information on receipts may need to be deleted.

What are boundaries, i.e. what changes cannot be facilitated:

- Architecture
    - The HBS backend can be deployed by other institutions, but any link to existing case management systems need to be developed by them. This includes synchronization across modes, when multiple modes are employed. A data collection strategy in which households can switch between modes within the reporting period or between household members would be problematic. When households use only one mode, then integration is relatively straightforward.
    - The current HBS does not allow for multiple questionnaires or a pre- and post-survey. This feature is currently under development. The questionnaire itself can be adjusted.
    - Allowing respondents to submit receipts by postal mail and use the app also or only in-house to enter receipts into the database is possible. However, a data file needs to be prepared carrying both user names and barcodes on respondent return envelopes.
- Methodology
    - Training and retraining of machine learning models or string matching routines for classification need to be set up from scratch, unless a table is available that uniquely transforms COICOP to the alternative classification.
    - A product list linking product names to the alternative classification may not yet be available, unless a table is available that uniquely transforms COICOP to the alternative classification.
    - Analyses scripts likely need to be revised because sample, paradata and auxiliary data files have changed.
- Logistics
    - When interviewers are involved, then the monitoring of incoming HBS data needs to be extended and intensified. In the ESSnet field tests, three reports were made per week and sent to coordinators. They then linked it to interviewer records and contacted interviewers about status and progress. This can be an extensive task.
    - While much can be learned from existing FAQ and known usability issues, the helpdesk function needs to be set up. This means helpdesk staff needs to be trained.
    - Monitoring demands may change, i.e. another institution may demand different data queries on top of existing queries. These need to be set up
    - The entire retraining procedure of machine learning models needs to be implemented at the institute, which may even demand for hiring or employing of data science expertise.

- Legal-ethical
    - Within household anonymization is not implemented. This was excluded as only a minority of countries deemed it important, but also because it poses methodological issues. When it is required, then the only current solution is to provide households with as many user names and passwords as needed. The backend does allow for grouping of user names belonging to the same household through the grouping option.
    - Respondents are encouraged to crop images and remove parts of the receipt that are irrelevant to the HBS. They are informed about this through an in-app instruction movie. However, they can ignore, forget or overlook and may submit receipts carrying potentially identifying information. Experience tells that such receipts are rare, but even so the risk exists. If it is demanded by legal constraints, then there is no other solution then to first ask respondents for explicit consent.

3.2 Zooming out: As part of a broader smart survey

Only changes are considered that are new with respect to section 3.1.

What may change?

- The products may need to be reported with (much) less detail.
- Only products with a certain minimum amount or in certain product categories may need to be reported.
- The purchase/expenditure data may supplement other survey questionnaires or even other smart surveys.
- Purchase/expenditure data may need to be integrated into other questionnaires, e.g. they may impact subsequent questions/data collections.
- The legitimization to collect detailed purchase data may be weaker/less apparent, i.e. it may just be a minor part of the information need.
- The reporting may stretch out over a much longer time period.
- Only purchases of one household member may be needed.

What are boundaries, i.e. what changes cannot be facilitated:

- Architecture
    - When the HBS survey is linked to other data collections, especially when those are dependent on data provided by the app, then the backend solutions need to be set integrated. The only solution would be to integrate questionnaires into the HBS app. Currently it is not possible to have a post-diary survey that asks questions based on what was reported in the app. This is also not planned. It is planned to have multiple questionnaires but these will be parallel and independent of in-app data.
    - Analysis services likely have to go through a full revision.
- Methodology
    - In-app instructions and tutorials may need to be changed when respondents are supposed to only report purchases with a certain minimum amount.
    - When certain products on receipts fall within the survey target scope and others do not, then the processing of receipts needs to be revised. Currently, the only solution is that respondents delete those while performing in-app editing of processed receipts.

    - When purchases below a certain amount are excluded, then the processing of data needs to be revised. For scan data entry currently the only solution is that respondents remove them by in-app editing. There are not yet in-app configurable edit rules on minimum amounts; only on negative amounts.
    - When certain products or product categories are left out, it will be advisable to adjust trained machine learning models. Knowing that classifications are no longer valid, would likely increase performance.
    - A very long reporting period would make the diary task too burdensome. Consequently, it may require that respondents report at a much lower frequency than daily. This would imply a new UI and data structure, i.e. a major revision of the app.
- Logistics
    - Most likely monitoring of incoming survey data will have to be changed.
    - When interviewers are employed and the app needs to be integrated into a broader data collection, then likely the support of interviewers will change.
- Legal-ethical
    - The ethical basis for inviting respondents to submit detailed expenditure data, i.e. at a higher level than actually needed, may become weak. This would imply that burden relative to gain may no longer be in balance and the app may not be used.

3.3 Zooming in: As part of a more specific smart survey

Only changes are considered that are new with respect to section 3.1.

What may change?

- A more detailed classifications of products is needed.
- Additional info on products is needed, e.g. yes/no ecological or quantities/metrics.
- Additional context is needed, e.g. whether products have been consumed, to how much calorie intake corresponds and/or purpose of the purchase.
- Details are needed on which household member consumed/used each product.
- The purchase time needs to be more precise than the calendar day.

What are boundaries, i.e. what changes cannot be facilitated:

- Architecture
    - Both the frontend and backend database need to be revised when additional questions are asked per purchased product and services.
    - Both the frontend and backend database need to be revised when reporting time slots are more detailed than calendar days.
    - More detailed classifications can be handled unless even the existing product lists are not detailed enough.
    - Analysis services need to be revised.
    - Additional smart features may be added such as photos or videos of consumed meals/products that require additional services to be added and the frontend and backend to be revised.
- Methodology
    - More detailed classifications require a new round of training of machine learning models. By using hierarchical classification, the additional training may be reduced.

    - More information and more context on products inevitably lead to survey questions to be asked on top of receipt processing. This requires additional usability testing and review of help and tutorial options.
    - Respondent burden and UI respondent interaction need to be reconsidered, in order to reduce the probability of break-off or lower data quality.
    - Additional smart features may be added such as photos or videos of consumed meals/products and need to be evaluated in terms of usability.
- Logistics
    - Monitoring will change.
    - Retraining of models through active and online learning may become more extensive.
- Legal-ethical
    - A new risk assessment and DPIA need to be made.

4. DISCUSSION

Modularity of the current HBS app has been discussed imagining three scenarios: as is, zooming out, and zooming in. Per scenario, the boundaries of modularity have been identified. Here, a last question is asked: What extensions and modifications can be made to increase modularity?

How can modularity be improved for HBS-type surveys (as is)?

- More elaborate documentation on backend deployment under a range of case management systems.
- Continue development of multiple questionnaires.
- Add configurability for when the app is used in-house.
- ESTAT-funded project @HBS2 develops an implementation plan, guidelines for shop and products lists and elaborated documentation on how to perform machine learning for classification.
- An interviewer access option, i.e. interviewers can see the status of cases assigned to them.
- Instruction material and FAQ for the helpdesk function can be formalized, e.g. through a dedicated website.
- Analysis queries and monitoring may be made more generic by allowing for uploading and programming of sample files, admin data files and fieldwork status file. Ideally, the app monitoring is integrated in general monitoring systems, but this may require a full integration with existing data collection channels and systems.
- An option may be added to personalize the app through individual passwords and individualized feedback or statistics.

How can modularity be improved when zooming out?

- Add also a post-survey questionnaire option where questions may depend on reported data in the diary.
- Make in-app help options and tutorials configurable, i.e. dependent on external parameters.
- Add edit rules to amounts and products to facilitate lower thresholds and selections of products.
- Create hierarchically trained machine learning models that are more efficient when required detail is smaller.
- Make the reporting time window configurable, e.g. days, weeks and months.

How can modularity be improved when zooming in?

- Embed the option to ask additional questions per reported product.
- Make the reporting time window configurable, e.g. hours, parts of the day and days.
- Construct hierarchical machine learning procedures such that additional classification levels can be added in a flexible and efficient way without having to discard higher levels.
- Search for additional existing data that may enrich product information, e.g. through EAN/GTIN product descriptions and known lists of ingredients.

Some of these actions are fairly easy and straightforward to implement, such as making the app configurable for in-house scanning, generalizing analysis services and personalized usage within a household, but most are more extensive. Inevitably, the decision to improve modularity is a trade-off in utility and costs.