A circular economy approach for lifecycles of products and services

Report on Validation and Evaluation of Traceability Solutions

Deliverable 5.4

PROJECT INFORMATION	
Type of Project	European Commission Horizon 2020
Topic	CIRC-01-2016-2017 Systemic, eco-innovative approaches for the circular economy: large-scale demonstration projects
Grant Agreement No.	776503
Project Duration	01/05/2018 – 30/04/2021 (36 months)
Project Coordinator	Nottingham Trent University (NTU)
Project Partners	Enviro Data (ENV), Jonathan Michael Smith (JS), Kosnic Lighting Limited (KOS), Centre of Research for Energy Resources and Consumption (CIR), European EPC Competence Center GmbH (EECC), The Institute for Ecology of Industrial Areas (IETU), RISE IVF AB (RISE), Make Mothers Matter (MMM), ONA PRODUCT (ONA), INDUMETAL Recycling (IND), GS1 Germany GMBH (GS1G), Laurea University of Applied Science (LAU), Center for European Policy Studies (CEPS), Institute of Communication and Computer Systems (ICCS), Recyclia (REC), S.A.T. Alia (ALIA)

DOCUMENT INFORMATION	
Title	D 5.4: Report on Validation and Evaluation of Traceability Solutions
Version	1.0
Release Date (dd/mm/yy)	20/12/2019
Work Package	WP5
Dissemination Level	PU

DOCUMENT AUTHORS AND AUTHORISATION	
Document Responsible	EECC
Contributors	EECC, ICCS
Reviewed by	GS1 Germany, CIRCE
Approved by	Prof. Daizhong Su, NTU

DOCUMENT HISTORY			
Version	Date (dd/mm/yy)	Description	Implemented by
0.1	16/12/2019	First draft	EECC
0.2	18/12/2019	Second draft added sections 6.5, 7	EECC
0.3	19/12/2019	Incorporating comments of CIRCE and RISE	EECC
0.4	19/12/2019	Incorporating comments of GS1G	EECC
1.0	20/12/2019	Final Document D5.4	EECC

Summary

This document reports on the evaluation and validation of the traceability solutions developed in work package WP 5 and is resulting from the work of Task 5.5.

For this purpose, 71 automated software test cases for the developed software artefacts have already been co-developed during implementation. Thus, the function of the software modules is automatically tested and verified at each development iteration.

In addition, the build process of the traceability solutions is fully automated. The same applies to the deployment of the software on the servers operated by a service provider in a data centre in Germany. The high degree of automation of this build and deployment chain ensures the highest level of reliability.

At the next higher stage, the integration testing between the individual Traceability Applications and on operational Level Application Containers of the traceability module, the interoperability was validated with automated testing methods in order to ensure the correctness of operation at the highest possible level.

In addition to these internal automated test stages, interoperability with the ICT platform was validated in integration tests. These tests are not automated and were therefore carried out in complete form for the first time at a face-to-face meeting in Cologne in October 2019 and repeated by ICCS and EECC for new releases.

Finally, the validation of the developed traceability tools with regard to their integrability into the CEBMs was successfully tested using test data and initial developments for CIRC4Life demo use cases. The traceability module developed in this work package is based on the currently valid GS1 standards "EPCIS 1.2" and "CBV 1.2". Additionally, extensions based on the drafts of the respective standards in versions 2.0 as they were described in D 5.1 are supported and tested. The traceability module is thus able to provide the traceability of products and materials as well as the data on their effects on the ecological footprint. This means, historical product information can be made available on an individual instance level for Life Cycle Assessment (LCA) calculations or other use cases arising from the new business models.

Table of Content

List of Figures:

List of Data:

Acronyms and abbreviations

Abbreviation	Description
CBV	Core Business Vocabulary for EPCIS
CEBM	Circular Economy Business Model
EPC	Electronic Product Code
EPCAT	Product Name of EPCIS Implementation by EECC
EPCIS	Electronic Product Code Information System
(S)GLN	GS1 Global Location Number (with Extension)
ID	Identifier
GTIN	GS1 Global Trade Item Number
LCA	Life Cycle Assessment
POS	Point of Sale
PU	Public, fully open, e.g. web
SGTIN	GS1 Serialised GTIN
TLS	Transport Layer Security
WP	Work Package
XSD	XML Schema Definition

1 Introduction

The aim of task 5.5 is the validation of the solutions developed in this work package.

The EPCIS-based traceability solutions developed have been deployed, tested and validated for the CIRC4Life environment, including three circular economy business models, ICT platform, etc. and case studies have been conducted to validate the tools provided.

The task included the following activities:

- Unit tests for all developed components to ensure correct technical functionality
- Deployment and operation of an EPCIS implementation (EPCAT), the core EPCIS modules and the traceability solutions for the new circular economy business models to enable for the following test setups:
    - Integration testing between different traceability components to validate correct functionality of the traceability solution
    - Integration testing between the traceability solutions and the ICT platform modules to validate correct interaction with the ICT platform
    - System testing with test data to validate that the overall system meets the requirements

2 Unit Tests

EECC has developed unit tests for all components to ensure correct technical functionality at the level of individual functions. All unit tests are run on every build, in particular as part of the release and deployment pipeline, hence the correct functioning of all parts is ensured.

Unit tests in particular ensure the following:

- All web service endpoints accept well formatted inputs (tested with exemplary data)
- All web services reject malformed calls or inputs and return appropriate http error codes and a meaningful error message
- All handlers for incoming data validate and transform the inputs as expected
- All internal data transformations work as expected (tested with exemplary data)
- All capturing applications produce well-formed and EPCIS standard conformal XML data for storage in the Traceability Module Core
- XML data for the new CIRC4Life EPCIS Impacts Extension is generated in the format
- specified by the XSD (EECC, 2019) as described in D5.2 by the corresponding capturing applications
- Impacts are extracted correctly from EPCIS data conforming to the Impacts extension (various examples tested)
- The business event models are implemented correctly, i.e.
    - Serialization yields EPCIS standard conformal events (XML well formed)
    - Standard EPCIS events are recognized as the correct CIRC4Life business events

The JUnit Framework (JUnit, 2019) is used for the test development and the tests are run by Maven (Apache Software Foundation, 2019) during the test phase as part of the build pipeline, see Figure 1. At the moment 71 unit test cases are implemented. The detailed test reports can be found in the appendix section 8.1.

3 Deployment and Operation

All Traceability tools are built on Java version 11. By using GitLab, Jenkins, and Maven, EECC has established an automated build pipeline that runs on EECC's servers in Germany. All code is accessible to EECC only. This pipeline automatically tests every change to the Traceability Module code and immediately reports on errors, such as failing unit tests. Deployment is automated by using the same stack, in particular Jenkins and Maven are used in order to compile in a stable environment. The core EPCIS modules are compiled into the capturing and accessing applications. After a successful release build, docker containers are automatically generated on the basis of the `openjdk:11-slim` docker image and distributed through EECC's own registry to the server running the applications. See Figure 1 for a schematic depiction of the build pipeline[*].

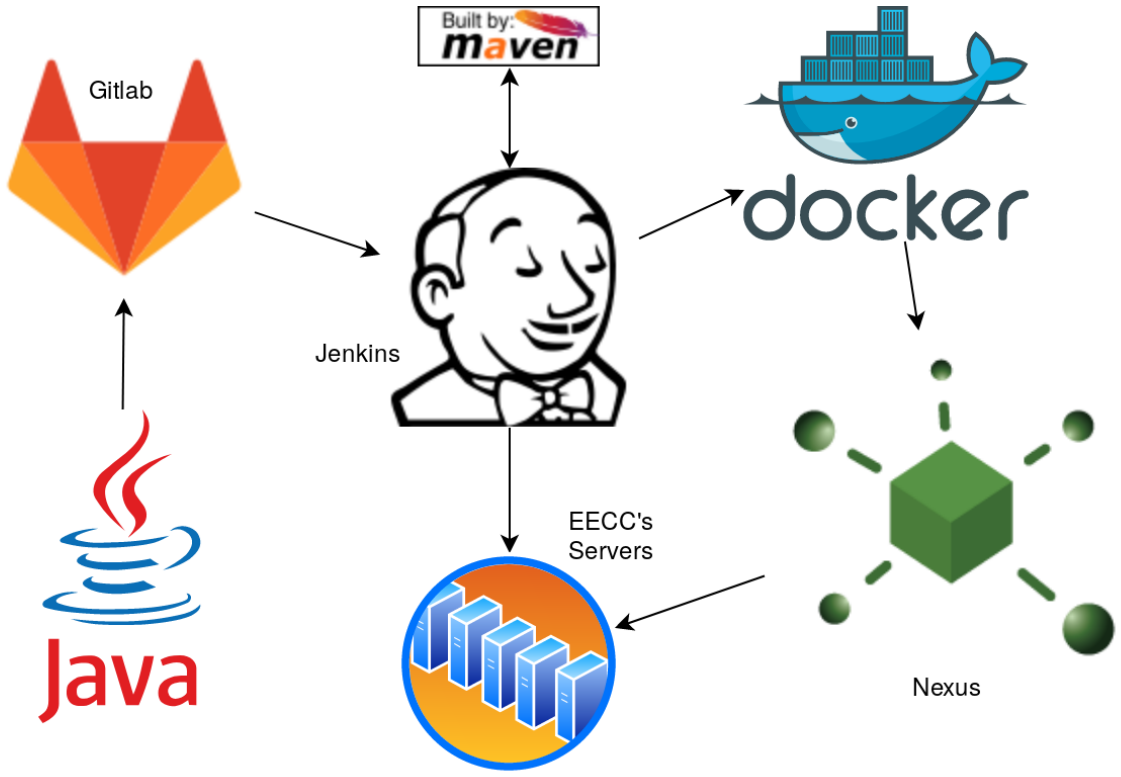


Figure 1: Schematic Depiction of the Traceability Module's Build pipeline

EECC's own EPCIS 1.2 certified event repository, EPCAT, is provided as a dockerized service (Docker, 2019) with a dedicated instance running for CIRC4Life in order to ensure segregation and confidentiality of all traceability data. The servers are monitored and extensive log-files with debugging information are generated in production in order to enable timely trouble shooting and debugging. Nginx is used as a reverse proxy for the Java Spring Boot based web applications which are running their own tomcat webservers in order to handle http requests. Let's Encrypt certificates are used for TLS encryption and all internet traffic to circ4life.eecc.info is encrypted. A redirect from port 80 (http) to 443 (https) ensures that users or connecting applications do not accidentally send login credentials over an unencrypted channel. Certbot (Certbot, 2019)is integrated with Nginx to facilitate

[*] A build pipeline is a sequence of jobs to be executed after new code has been checked in to the code repository. These include compile and build, test as well as deployment on the servers

regular certificate (Let's encrypt, 2019) renewal. See Figure 2 for a depiction of service calls and their routing through the traceability Module's components.

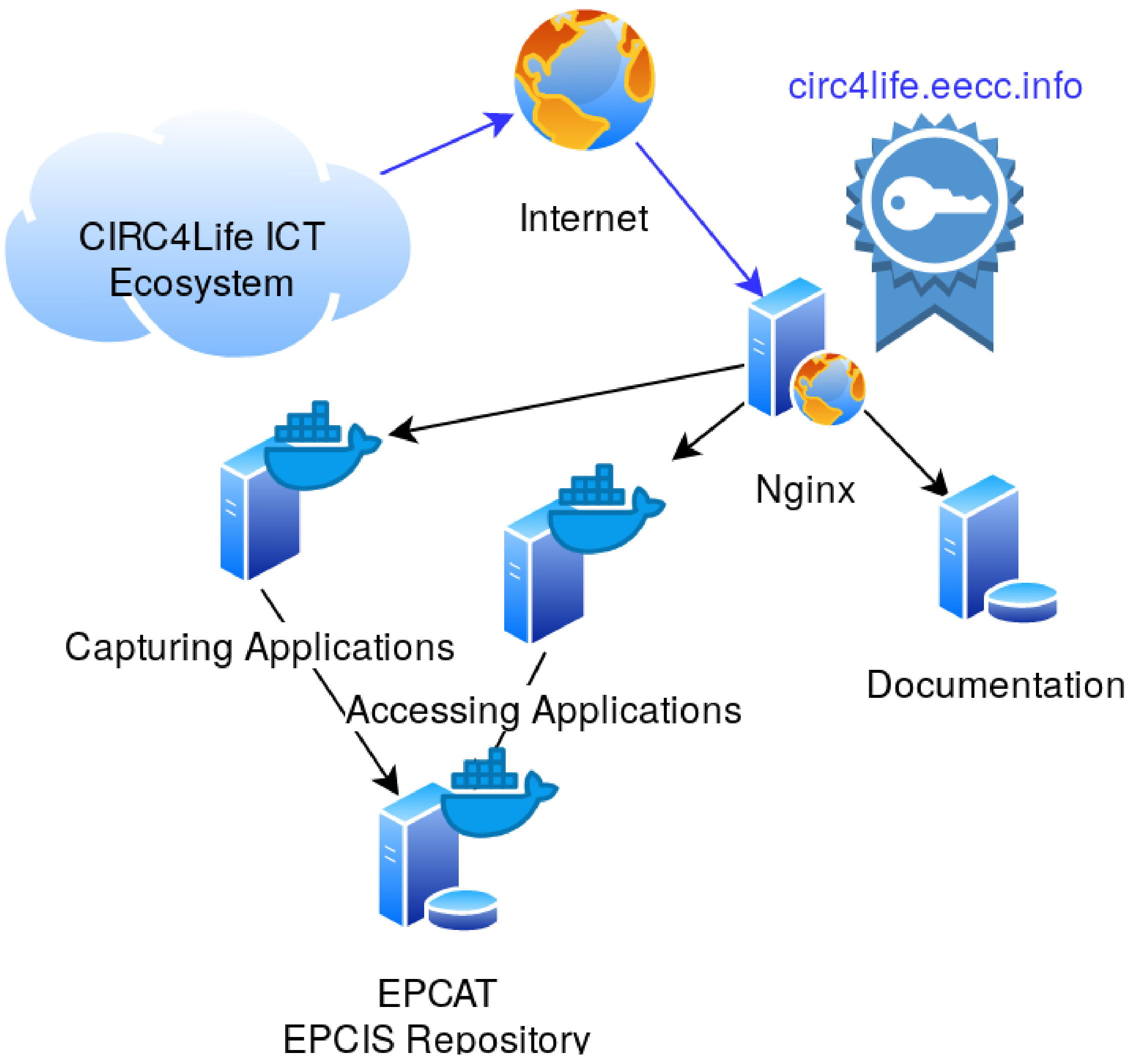


Figure 2: Schematic Calls to Traceability Tool's Resources through the Dockerized Web Services.[*]

[*] Arrows represent http calls. Encrypted connections are depicted in blue.

4 Integration Testing Between Different Traceability Components

4.1 Integration Tests within Traceability Applications

Integration tests in the narrower sense, i.e. between different classes of the same application, are executed on every build using the same framework as the unit tests described in Section 1.

The integration tests ensure that

- The core classes which are shared by accessing and capturing applications work well with both. These include in particular
    - The business events models
    - Converters that perform data transformation
    - Utilities
- The aggregation modules inside the accessing applications work as expected
- The event handlers work with the subscription accepting module of the accessing application as expected
- The master data module is working as expected
- All Capturing Controllers transform data as expected
- All data handling class hierarchies are properly constructed and functioning

The detailed test reports can be found in the Appendix section 8.1

4.2 Integration between Traceability Application Containers

Figure 2 shows that internal http communication between different components of the Traceability Module, concretely different containerized spring boot applications, is part of the design. The Capturing and Accessing Applications communicate with the EPCIS repository by using the standard EPCIS SOAP interface, hence tests of the integration of these components are important to ensure the overall functioning of the Traceability Solutions.

For the scope of CIRC4Life, we call such tests between different containers of the Traceability Module "Integration Tests", although such tests involving different Applications might already be called system tests in other environments. Consequently, the implementation of the test to ensure proper functioning of the whole setup in the production environment is a little more complex than the Unit tests mentioned in Section 1.

EECC has implemented automated testing of the docker environment in the following way:

- On each deployment, each application sends a specific (EPCIS) test event to the core repository (EPCAT)
- The capturing application queries its own test event and compares versions in order to ensure that the test event was successfully captured

- The accessing application queries its own event in order to ensure that querying EPCAT works as expected
- The accessing application additionally checks that it received the first as well as the latest test event from EPCAT through its subscription in order to verify that the subscription is working and that the complete event history has been received and processed.

These tests are run immediately after deployment. There are appropriate re-try loops and timeouts to allow for delays in the applications start-up procedures. If any of these Traceability Module self-tests fails eventually, the deployment is considered to have failed and the system is rolled back to the last working version. To this end, all versions of the docker containers are kept in EECC's registry at least until a newer version was found to be working.

5 Integration testing between the traceability solutions and the ICT platform modules

Integration with the ICT platform means that the Traceability Module applications are exchanging data with the applications developed by ICCS via the Internet. Concretely, all those applications are web applications running as servers and data exchange happens over TLS encrypted http connections. Mutual identification of the web applications is done by pre shared secrets (passwords) given out by ICCS. The credentials are checked by the Security Module, which is a KeyCloak Server (RedHat, 2019) deployed and operated by ICCS. According to the OpenID Connect layer (OpenID, 2019) on top of the OAuth 2.0 protocol (OAuth, 2019), a bearer token is given out upon successful identification by the Security Module to the requesting application. The Security Module than authorizes access to each module based on the identified roles.

After a secured connection has been established and authorised access has been granted, the actual data exchange through the RESTful web services happens in JSON format. All endpoints and data models of the ICT platform as well as the Traceability Module are described using open API specifications which yields a human as well as machine readable documentation. Test protocols have been developed and executed manually to ensure that the services work together as planned (see D 5.1, D 5.2, D 5.3, D 4.1, D 4.2), in particular

- The data exchange formats match the specifications
- Data is transformed as expected on both ends
- The returned http status codes are es expected in both success and error cases
- The overall data flow goes forth and back in the expected pattern

The test protocol can be found in the appendix section 8.2. It was successfully carried out during the technical meeting in Cologne on the 22nd and 23rd of October and repeatedly after new ICT platform releases.

6 System testing with test data

It has been validated that the overall Traceability System meets the requirements which have been defined in Task 5.1 and adapted to the needs of the CEBMs and demonstrations in pace with their respective development.

6.1 Recycling and Reuse

This section contains a description of a full system test for all processes of the recycling demonstration in which the Traceability Module is involved. The concrete example is shown for tablets, but the processes for other EEE items are identical. Very similar processes are planned for bio-waste and will be handled by the Traceability Module in the same way. The data flow for the Recycling CEBM in its latest version has been described in D5.3. It is depicted again in Figure 4.

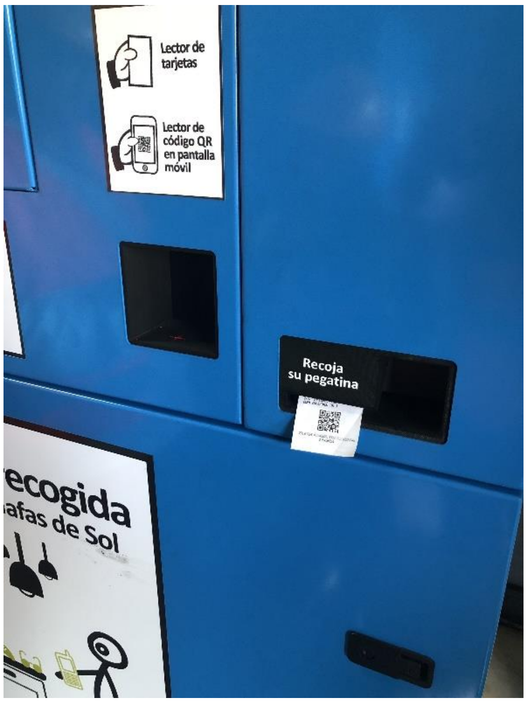


Figure 3: Picture of the intelligent bin with a barcode sticker used for CEBM collaborative recycling

6.1.1 Disposal Event

According to Figure 4, the first interaction with the Traceability Module in the recycling demonstrations is a message from the intelligent bin's control unit, as developed by NTU. This is a very simple data package. An example is given in Data 1.

```json
{
  "binBarcode": "160040000011811220000011",
  "eventTime": "2019-05-21T14:02:00+02:00",
  "fillingLevel": 50.5
}
```

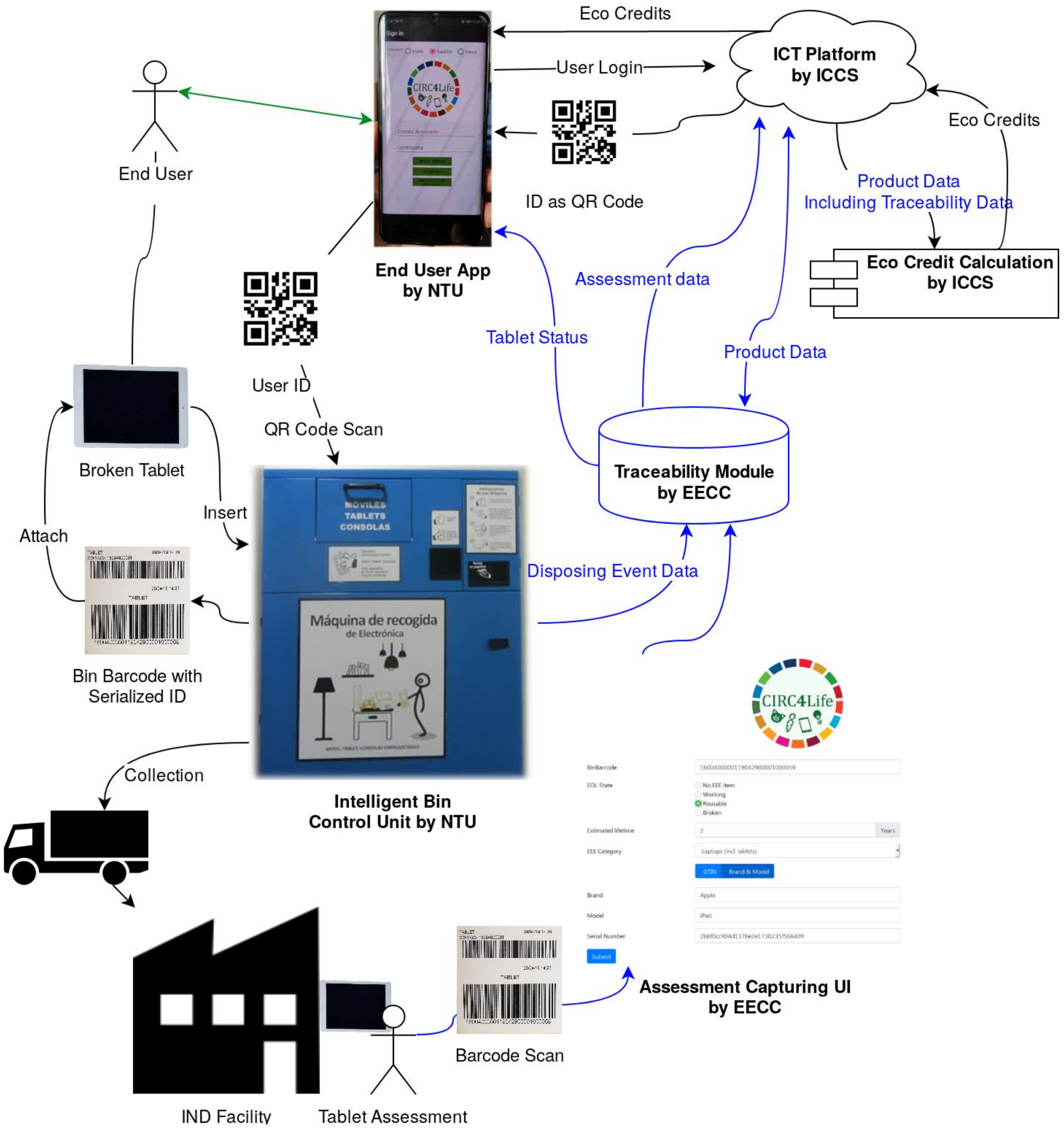
Data 1: Example for Test Input Data to the **`Bin Disposal Event Receiving Endpoint`**

Using e.g. the Swagger-UI provided by the Traceability Module, sending data through the API endpoints can be tested any time. The capturing endpoints testing UI can be found at `https://circ4life.eecc.info/swagger-ui.html` and provides a section for testing the **`Bin Disposal Event Receiving Endpoint`**. Notice that after finishing the test phase, all endpoints will be access protected. The expected response for sending the above request is a status code 200 (OK) and a JSON with details about the success or failure as appropriate. An example is given in Data 2.

```
{
  "timestamp": "2019-12-12T10:20:41.777588",
  "status": 200,
  "message": "
      <?xml version=\"1.0\" encoding=\"UTF-8\" standalone=\"yes\"?>
      <captureResult>
         <cntCapturedEvents>1</cntCapturedEvents>
         <durationInMs>7</durationInMs>
         <startTime>2019-12-12T10:20:41.769Z</startTime>
         <endTime>2019-12-12T10:20:41.776Z</endTime>
         <capturedEventIds>
            <eventId>4f5be11c-593b-4b39-8a9a-451c796160f2</eventId>
         </capturedEventIds>
         </captureResult>
        ",
  "path": "/api/capture/bin-disposal-event",
  "reason": "OK"
}
```

Data 2: Example response of the **`Bin Disposal Event Receiving Endpoint`**[*].

[*] Here, the message included is the EPCIS conformal answer of the central repository confirming that the event was stored.



BinBarcode	160040000011904290001000059
EOL State	No EEE item Working [✓ Reusable] Broken
Estimated lifetime	2 Years
EEE Category	Laptops (incl. tablets)
	GTIN Brand & Model
Brand	Apple
Model	iPad
Serial Number	2b6f0cc904d137be2e1730235f566409
Submit	

Figure 4: Detailed View on the Recycling Data Flow and ICT Systems Interaction.[*]

EECC uses a few example data packages of the form of JSON Data 1 to automatically test the correct operation of the endpoint.

[*] This is Figure 13 of D 5.3

6.1.2 Tablet Status while in the Bin

If the end user queries the tablet status through the mobile App developed by NTU, the Traceability Modules **`Item Status Endpoint`** is called. The Swagger-UI to test all accessing endpoints can be found at https://circ4life.eecc.info/access/swagger-ui.html. Note that this is access protected, in order to ensure that all traceability data collected within CIRC4Life is kept confidential, even during the testing period.

When querying the status of the item that was deposited when generating Data 1, i.e. for which the bin prints a barcode label encoding the number 16004000001181122000011, the answer is expected to be similar to the one shown in Data 3.

```
{
  "epc":
"https://circ4life.eecc.info/recycling/bin/obj/16004000001181122000011",
  "disposition": "urn:epcglobal:cbv:disp:returned",
  "location": "https://circ4life.eecc.info/recycling/loc/00001"
}
```

Data 3: Example for the data returned by the **`Item Status Endpoint`**, when queried for an item that is currently in the bin.

The short answer returned in Data 3 is an excerpt from the EPCIS standard events. The latest known location (a recycling bin ID custom to CIRC4Life in EPCIS conformal format) and disposition (using EPCIS core business vocabulary) are returned.

The meaning of typical status values for Circ4Life is the following. The status URI always starts with **`urn:epcglobal:cbv:disp`**: followed by:

`.returned`	Item is returned into a smart recycling bin and awaiting collection
`.sellable_not_accessible`	Item is in working condition and will be reused
`.damaged`	Item is damaged and needs to be repaired before it can be reused
`.disposed`	Item is broken and will be recycled
`.unknown`	An item which was put into a smart bin was found to be not of a recyclable type according to the OAS (EECC 2019b).

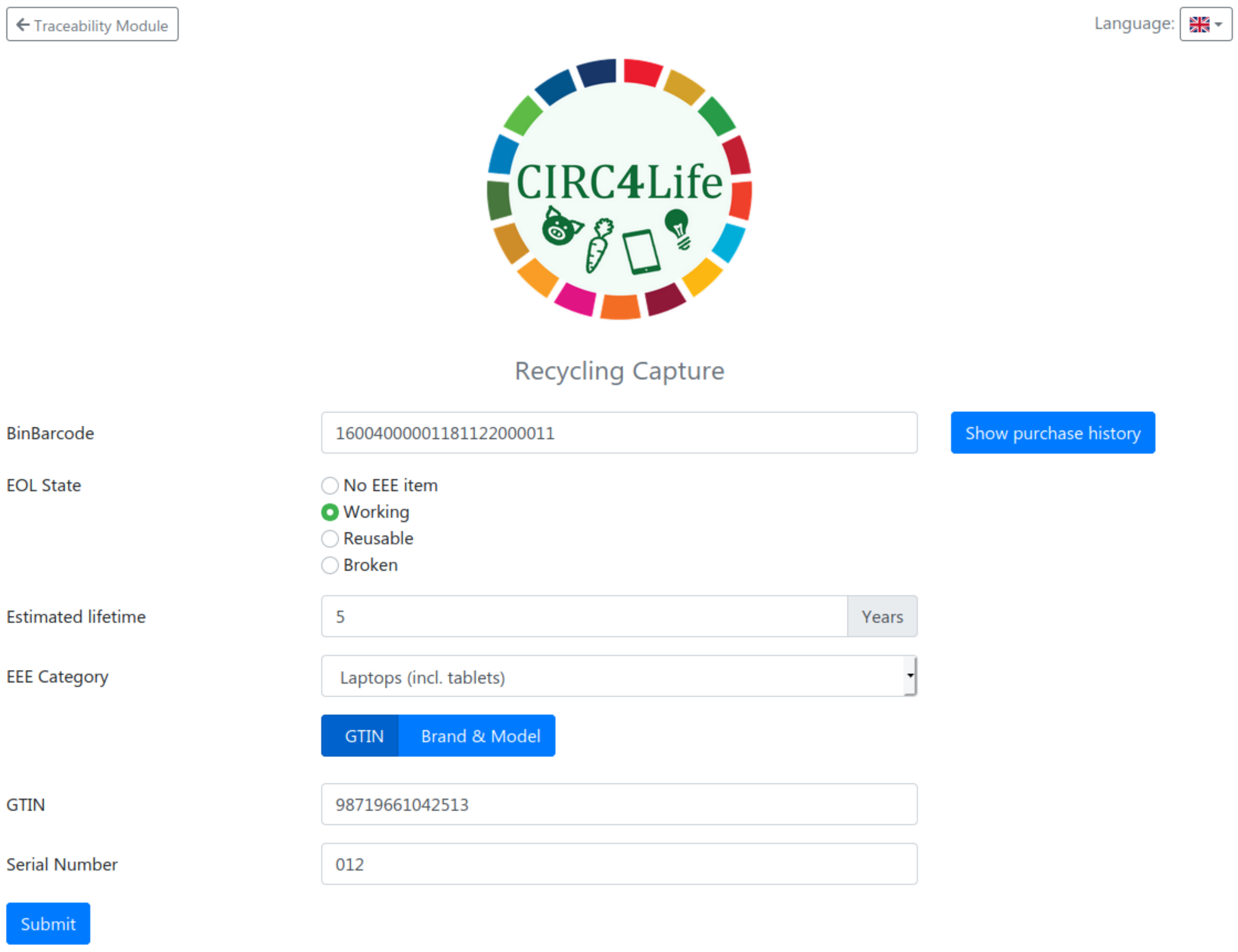
6.1.3 EEE Assessment

After collection, the tablet is assessed at Indumetal's facility and the outcome of the assessment entered into the graphical user interface of the assessment capturing application. SeeFigure 5 or a screenshot. If the data is entered as in the screenshot (i.e. the tablet is found to be

working), a subsequent call to the Item Status Endpoint will yield Data 4.

```
{
  "epc":
"https://circ4life.eecc.info/recycling/bin/obj/16004000001181122000011",
  "disposition": "urn:epcglobal:cbv:disp:sellable_not_accessible",
  "location": "urn:jaif:id:loc:25LUN466724325"
}
```

Data 4: Example for the data returned by the `Item Status Endpoint`, when queried for an item that has been assessed at Indumetal's facilities and found to be working[*].



Recycling Capture

BinBarcode	16004000001181122000011
EOL State	No EEE item [✓ Working] Reusable Broken
Estimated lifetime	5 Years
EEE Category	Laptops (incl. tablets)
GTIN	98719661042513
Serial Number	012

Figure 5: Screenshot of the graphical web user interface for capturing assessment data from the recycler

6.1.4 Using the User's Purchase History for Automatic Completion

In Figure 5 a *Show purchase history* button has appeared in the UI after the barcode has been scanned. This button only appears, if the user has a purchase history in his eco-account, i.e. if the customer used the eco shopping application to track the footprint of his buying behaviour. The recycler can view this purchase history and select the recycled item from there, if present. In this case, the product information in the form is automatically filled. See Figure 5.3 for a screenshot.

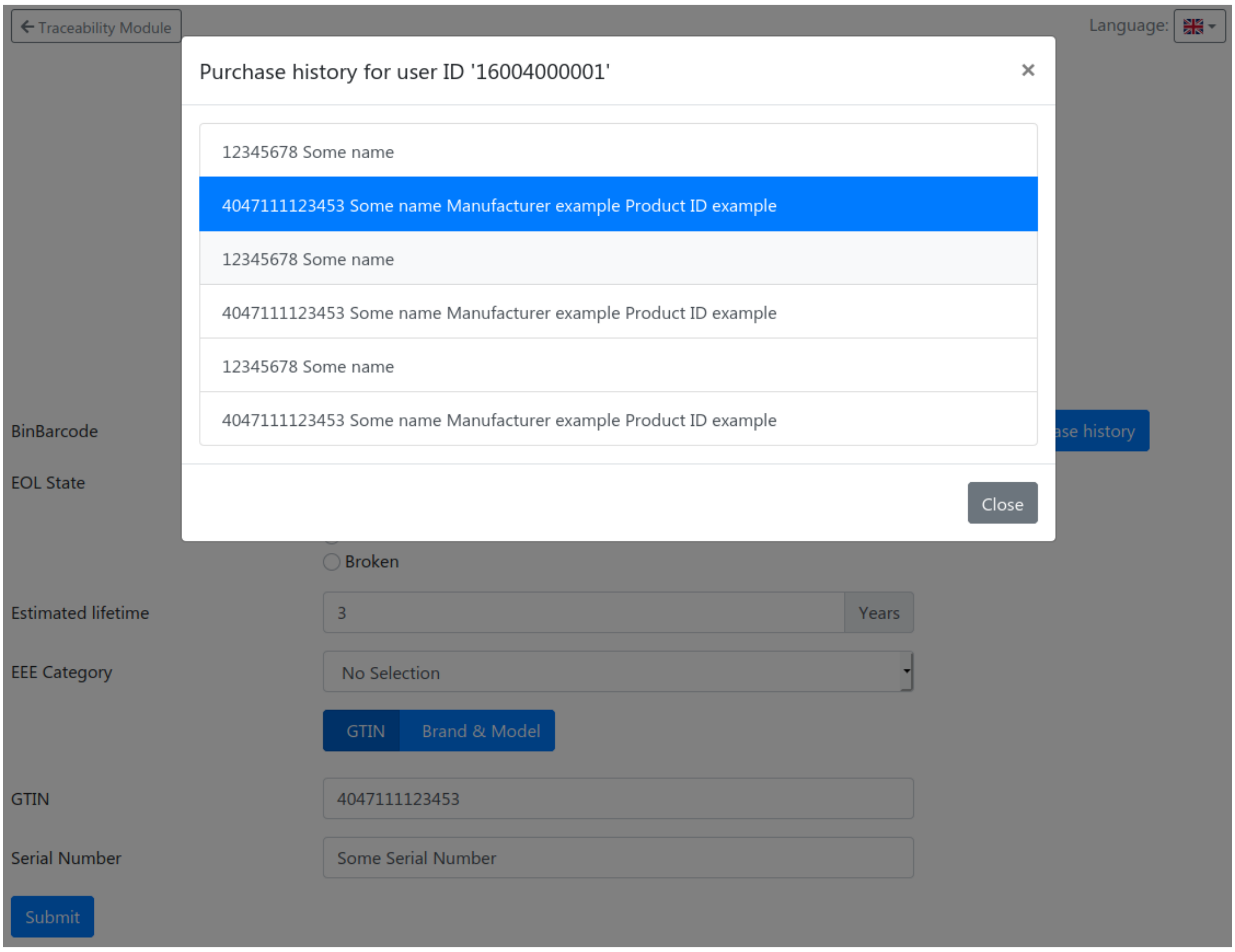
The purchase history is received from the ICT Platform via a dedicated endpoint. An example is given in Data 5.

```
[
```

[*] Since Indumetal does not own a GLN, a location identifier is constructed from the DUNS.

```
  {
    "gtin": 12345678,
    "manufacturer": "",
    "model": "",
    "name": "Some name",
    "serialnum": "Some Serial Number",
    "type":
    {
      "hskey": -1,
      "unukey": null,
      "text": null
    },
    "purchasedate": "2017-01-01"
  },
  {
    "gtin": 4047111123453,
    "manufacturer": "Manufacturer example",
    "model": "Product ID example",
    "name": "Some name",
    "serialnum": "Some Serial Number",
    "type":
    {
      "hskey": 847130,
      "unukey": null,
      "text": null
    },
    "purchasedate": "2016-03-02"
  }
]
```

Data 5: Example of a user's purchase history data as produced by ICCS



Purchase history for user ID '16004000001'

12345678 Some name
4047111123453 Some name Manufacturer example Product ID example
12345678 Some name
4047111123453 Some name Manufacturer example Product ID example
12345678 Some name
4047111123453 Some name Manufacturer example Product ID example

BinBarcode
EOL State
Broken
Estimated lifetime: 3 Years
EEE Category: No Selection
GTIN | Brand & Model
GTIN: 4047111123453
Serial Number: Some Serial Number

Figure 6: Screenshot of the graphical web user interface of the Traceability Module showing the dialog to select an item from the users purchase history. ICCS example data for the purchase history is visible.

6.1.5 Recycling Evaluation Push

After the Capturing Application has stored the inspection event in the central EPCIS repository, the event is sent to the accessing application due to its subscription. The accessing application recognizes the inspection event, transforms the data into an evaluation report as specified by ICCS, and sends the data to the ICT Platform in order to have the eco-credits calculated for the user. See Data 6 and 7 for an Example of an item for which the GTIN is known to the recycler.

```
{
  "consumer": 16004000001,
  "report":
  {
    "recycledate": "2019-12-13",
    "lifetime":
    {
      "years": 5,
      "extramonths": 0
    },
    "item":
    {
      "wastetype": 847130,
      "product":
      {
        "gtin": 4250236813301,
        "serialnum": "012",
        "name": "LOOKUP BY GTIN",
        "type":
        {
          "hskey": 847130,
          "unukey": 303,
          "text": "Laptops (incl. tablets)"
        },
        "purchasedate": "2014-12-14"
      }
    },
    "state": "WORKING"
  }
}
```

Data 6: Example for an evaluation report send to ICCS.

```

{
  "consumer": 16004000001,
  "report":
  {
    "recycledate": "2019-12-14",
    "lifetime":
    {
      "years": 5,
      "extramonths": 0
    },
    "item":
    {
      "wastetype": 847130,
      "product": null
    },
    "state": "WORKING"
  }
}
```

Data 7: If no GTIN is known to the recycler, the report cannot contain an item reference (“product”) according to the OAS of the ICT Platform. Data of this form will still yield eco-credits to the user derived from lifetime and waste type.

6.2 Generic Capturing Services

As explained in the earlier deliverables of WP5, the Traceability Module provides generic interfaces to capture impacts for an item without associating the impact with a specific business step. This is not optimal in terms of traceability, since e.g. analysing, which processes cause the largest impacts might be difficult, if a lot of impact sources are unknown. However, having an aggregated, general list of impacts for a product or parts of a complex product from a supplier might be sufficient and is certainly preferable to not tracking those impacts at all. Furthermore, due to the volatile requirements, these endpoints provide a fallback for capturing impacts from processes which were not envisioned during task 5.1. Anyhow, the traceability tools provide software endpoints to gather impacts wherever needed. All these capturing endpoints are documented and can be tested at **`https://circ4life.eecc.info/swagger-ui.html`**. Examples for all capturing endpoint inputs and returns can be found there as well.

6.2.1 Generic Impacts Capturing

Impacts can be recorded for any item by pushing data similar to the Data 8 example to the Generic Impacts Capturing Endpoint, naming the EPC of the product causing the impacts in the path. In this example and throughout, the usage of semantic web vocabulary* is promoted. Existing vocabulary should be used whenever appropriate and EPCIS conformal identifiers exist. It is also shown how custom identifiers can be built if no appropriate vocabulary is known. A typical response is given in Data 9.

```

{
  "resourceList":
  {
    "epcList": [
"https://circ4life.eecc.info/recycling/bin/obj/12345678901190514000011123 45"
    ],
    "quantityList": [
      {
        "epcClass": "http://circ4life.eecc.info/material/copper",
        "measure":
        {
          "quantity": 0.5,
          "uom": "KGM"
        }
      }
    ]
  },
  "wasteList":
  {
    "resourceOrWasteList": [
      {
        "type": "http://circ4life.eecc.info/material/plastics",
        "amount":
        {
          "quantity": 0.5,
          "uom": "KGM"
        },
        "sourceOrSink": "http://circ4life.eecc.info/sink/landfill"
      }
    ]
  },
  "transportationList": [
    {
```

* Example for semantic web: In databases for books one database may use the term "author", whereas the other may use the term "creator". To make the integration complete, and extra definition should be added to the RDF (Recource description Framework) data, describing the fact that the relationship described as "author" is the same as "creator". This extra piece of information is, in fact, a vocabulary (or an ontology).

```
        "vehicle": "http://purl.org/vso/ns#Truck",
        "distance":
        {
          "quantity": 10,
          "uom": "KMT"
        }
      }
    ],
    "impactList": [
      {
        "impactType": "http://circ4life.eecc.info/type/land_usage",
        "measure":
        {
          "quantity": 1,
          "uom": "KMK"
        }
      }
    ]
}
```

Data 8: Example input for the **`Generic Impacts Capturing Endpoint`**[*].

```
{
    "timestamp": "2019-09-15T11:29:16.333482",
    "status": 200,
    "message": "Impacts recorded successfully.",
    "path":
"/api/capture/impact/urn%3Aepc%3Aid%3Asgtin%3A4047111.012345.012345678901",
    "reason": "OK"
}
```

Data 9: Example answer of the Generic Impacts Capturing Endpoint.

6.2.2 Generic Events Capturing

The **`Object Event Capturing Endpoint`** can be used to capture any type of object event. This is a JSON interface to the underlying EPCIS 1.2 conformal SOAP interface. An example input is given in Data 10.

```
{
    "action": "OBSERVE",
    "epcList": [
      "urn:epc:id:sgtin:4047111.012345.012345678902"
    ],
    "disposition": "urn:epcglobal:cbv:disp:returned",
    "readPoint": "https://circ4life.eecc.info/recycling/loc/00001",
    "bizLocation": "https://circ4life.eecc.info/recycling/loc/00001",
    "bizStep": "urn:epcglobal:cbv:bizstep:arriving",
    "eventTime": "2019-08-15T14:33:00+02:00",
    "impactData":
```

[*] containing an example for resources being used (a concrete recycled item and 0.5 kg of copper), waste being produced (0.5 kg of plastics which go to landfill), a transportation (10 km by truck) and a generic impact (1 km^2 of land usage)

```
    {
      "resourceList":
      {
        "quantityList": [
          {
            "epcClass": "https://w3id.org/saref#Electricity",
            "measure":
            {
              "quantity": 0.5,
              "uom": "WHR"
            }
          }
        ]
      },
      "wasteList":
      {
        "resourceOrWasteList": []
      },
      "transportationList": [
        {
          "vehicle": "http://purl.org/vso/ns#Truck",
          "distance":
          {
            "quantity": 50,
            "uom": "KMT"
          }
        }
      ],
      "impactList": []
    },
    "quantityList": [],
    "bizTransactionList": [],
    "sourceList": [],
    "destinationList": [],
    "ilmd": []
  }
```

Data 10: Example input for the **`Object Event Capturing Endpoint`**[*].

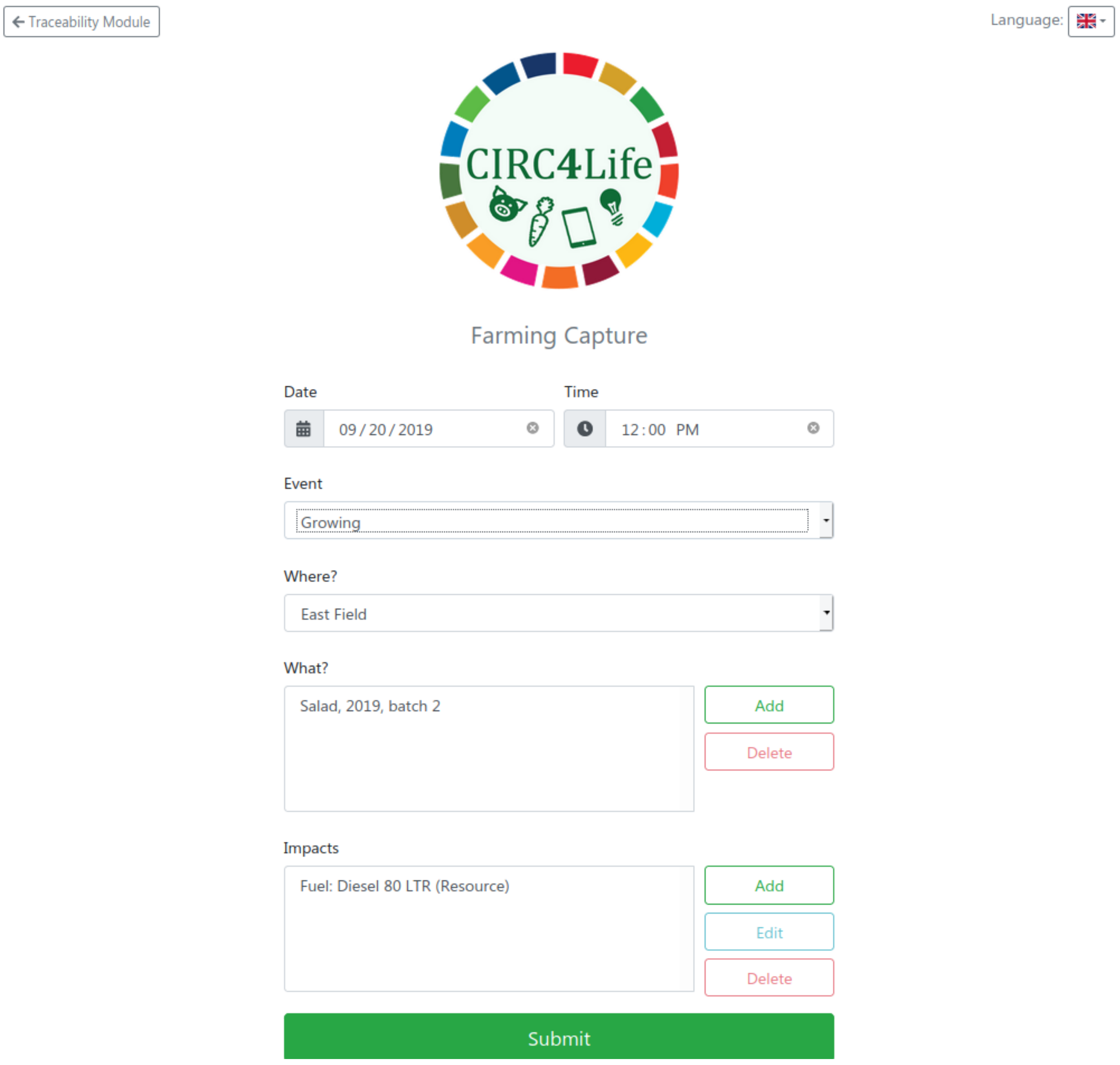
6.3 Supply Chain Events Capturing

6.3.1 Vegetables

A prototype of a graphical user interface has been developed for capturing product batch specific information about impacts of farming activities, such as soil preparation, planting, weeding, fertilizing, watering, harvesting and also recycling (composting). See Figure 7 for a screenshot. The interface is already fully functional (see Data

[*] The event represents a bin disposal event as in Section 6.1.1, but additionally recording impacts due to transportation and electricity consumption. Some empty elements where added here in order to indicate further optional fields.

11 for an example event), but the further development and in particular opportunities to integrate with the Farm Carbon Calculator are currently being discussed between JS and EECC.



Farming Capture

Date	09/20/2019
Time	12:00 PM
Event	Growing
Where?	East Field
What?	Salad, 2019, batch 2
Impacts	Fuel: Diesel 80 LTR (Resource)

Figure 7: Screenshot of the graphical web user interface of the Traceability Module's farming data capturing application

```
<ObjectEvent>
      <eventTime>2019-09-20T12:00:00.000Z</eventTime>
      <recordTime>2019-12-12T10:12:10.984Z</recordTime>
      <eventTimeZoneOffset>+00:00</eventTimeZoneOffset>
      <epcList>
            <epc>https://www.scillyorganics.com/epcis/obj/salad.2019.2</epc>
      </epcList>
      <action>OBSERVE</action>
      <bizStep>urn:epcglobal:cbv:bizstep:repairing</bizStep>
      <disposition>urn:epcglobal:cbv:disp:active</disposition>
      <bizLocation>
```

```
            <id>https://www.scillyorganics.com/epcis/loc/east_field</id>
      </bizLocation>
      <c4l:impactList xmlns:c4l="https://circ4life.eecc.info/epcis">
            <c4l:impact>
                  <c4l:impactType>Fuel: Diesel</c4l:impactType>
                  <c4l:measure>
                        <c4l:quantity>80</c4l:quantity>
                        <c4l:uom>LTR</c4l:uom>
                  </c4l:measure>
            </c4l:impact>
      </c4l:impactList>
      <eecc:eventId>7c8c0b5a-510a-44a0-b235-2ad2142a141d</eecc:eventId>
</ObjectEvent>
```

Data 11: Example for the EPCIS event generated by the farming data capturing application from the data in Figure 7.

6.3.2 Industrial Lighting

The example Data 12 shows an event where components are permanently assembled into a new product. Such Transformation events represent permanent assemblies from the production phase, like baking bread from the ingredients. This is not revisable. Due to this, in CIRC4Life EECC promotes to think in terms of aggregation rather than transformation events. The connotation for aggregation and also the EPCIS standard semantic here is different in that the aggregation is not permanent and parts can be removed again or exchanged (Data 13).

```
<TransformationEvent>
      <eventTime>2017-08-09T05:50:10.124Z</eventTime>
      <eventTimeZoneOffset>+02:00</eventTimeZoneOffset>
      <inputEPCList>
            <epc>urn:epc:id:sgtin:4047111.000345.1323</epc>
            <epc>urn:epc:id:sgtin:4047111.002445.3127</epc>
            <epc>urn:epc:id:sgtin:4047111.000366.1534</epc>
      </inputEPCList>
      <outputEPCList>
            <epc>urn:epc:id:sgtin:4047111.012345.012345678901</epc>
      </outputEPCList>
      <bizStep>urn:epcglobal:cbv:bizstep:commissioning</bizStep>
      <disposition>urn:epcglobal:cbv:disp:active</disposition>
      <bizLocation>
            <id>urn:epc:id:sgln:4047111.12345.0001</id>
      </bizLocation>
      <c4l:resourceList>
            <c4l:quantityElement>

      <epcClass>urn:epc:class:lgtin:4012345.011111.4444</epcClass>
                  <quantity>10.2</quantity>
                  <uom>KGM</uom>
            </c4l:quantityElement>
            <c4l:resourceElement>
                  <!-- This is an example for referring to an external
ontology. We recommend using an existing ontology within the semantic web
whenever possible.-->

      <c4l:resource>https://w3id.org/saref#Electricity</c4l:resource>
                  <c4l:amount>
                        <c4l:quantity>0.1</c4l:quantity>

                        <!-- For the table of codes, see "Recommendation No. 20: CODES FOR UNITS OF MEASURE USED IN INTERNATIONAL TRADE" by the UNITED NATIONS ECONOMIC COMMISSION FOR EUROPE-->
                        <c4l:uom>WHR</c4l:uom>
                  </c4l:amount>
                  <c4l:source>http://semanco02.hs-albsig.de/repository/ontology-releases/eu/semanco/ontology/SEMANCO/SEMANCO.owl#Not-Renewable_Energy_Source</c4l:source>
            </c4l:resourceElement>
      </c4l:resourceList>
      <c4l:wasteList>
            <c4l:wasteElement>

      <c4l:waste>http://circ4life.eecc.info/material/plastics</c4l:waste>
                  <c4l:amount>
                        <c4l:quantity>0.5</c4l:quantity>
                        <c4l:uom>KGM</c4l:uom>
                  </c4l:amount>

      <c4l:sink>http://circ4life.eecc.info/sink/landfill</c4l:sink>
            </c4l:wasteElement>
      </c4l:wasteList>
</TransformationEvent>
```

Data 12: Example for an assembly event which creates a new product[*].

```
<ObjectEvent>
      <eventTime>2019-04-08T16:07:56Z</eventTime>
      <eventTimeZoneOffset>+02:00</eventTimeZoneOffset>
      <epcList>
            <epc>urn:epc:id:sgtin:4047111.012345.012345678901</epc>
            <epc>urn:epc:id:sgtin:4047111.012345.012345678905</epc>
      </epcList>
      <action>OBSERVE</action>
      <bizStep>urn:epcglobal:cbv:bizstep:replacing</bizStep>
      <disposition>urn:epcglobal:cbv:disp:active</disposition>
      <bizLocation>
            <id>urn:epc:id:sgln:0614141.07346.1234</id>
      </bizLocation>
      <c4l:resourceList>
            <epc>urn:epc:id:sgtin:4047111.012345.012345678905</epc>
      </c4l:resourceList>
      <c4l:wasteList>
            <epc>urn:epc:id:sgtin:4047111.012345.012345678901</epc>
      </c4l:wasteList>
      <c4l:transportList>
            <c4l:transportation>
                  <c4l:vehicle>http://purl.org/vso/ns#Truck</c4l:vehicle>
                  <c4l:distance>
                        <c4l:quantity>10</c4l:quantity>
                        <c4l:uom>KMT</c4l:uom>
                  </c4l:distance>
            </c4l:transportation>
      </c4l:transportList>
</ObjectEvent>
```

[*] e.g. a lamp, see `outpuEPCList`) from parts (`inputEPCList`).

Data 13: Example for a replacing event which denotes replacing one item with another one.

```
<AggregationEvent>
      <eventTime>2018-08-08T08:16:25Z</eventTime>
      <recordTime>2018-08-10T14:43:39.345Z</recordTime>
      <eventTimeZoneOffset>+02:00</eventTimeZoneOffset>

      <parentID>urn:epc:id:sgtin:4047111.012345.012345678901</parentID>

      <childEPCs>
            <epc>urn:epc:id:sgtin:4047111.000345.1323</epc>
            <epc>urn:epc:id:sgtin:4047111.002445.3127</epc>
            <epc>urn:epc:id:sgtin:4047111.000366.1534</epc>
      </childEPCs>

      <action>ADD</action>
      <bizStep>urn:epcglobal:cbv:bizstep:commissioning</bizStep>
      <disposition>urn:epcglobal:cbv:disp:active</disposition>
      <bizLocation>
            <id>urn:epc:id:sgln:4047111.12345.0001</id>
      </bizLocation>
</AggregationEvent>
```

Data 14: This is an aggregation event which contains the same data (impacts omitted for brevity) as Data 12, but in this case the event denotes a temporary assembly.

The model supports even the exchange of components in a Lamp. This can be depicted by the Aggrgation/delete and Aggregation/add combination as described in Data 15.

```
<AggregationEvent>
      <eventTime>2018-09-08T08:16:25Z</eventTime>
      <recordTime>2018-09-10T14:43:39.345Z</recordTime>
      <eventTimeZoneOffset>+02:00</eventTimeZoneOffset>

      <parentID>urn:epc:id:sgtin:4047111.012345.012345678901</parentID>

      <childEPCs>
            <epc>urn:epc:id:sgtin:4047111.000345.1323</epc>
      </childEPCs>

      <action>DELETE</action>
      <bizStep>urn:epcglobal:cbv:bizstep:repairing</bizStep>
      <bizLocation>
            <id>urn:epc:id:sgln:4047111.12345.0001</id>
      </bizLocation>
</AggregationEvent>

<AggregationEvent>
      <eventTime>2019-08-08T08:16:25Z</eventTime>
      <recordTime>2019-08-10T14:43:39.345Z</recordTime>
      <eventTimeZoneOffset>+02:00</eventTimeZoneOffset>

      <parentID>urn:epc:id:sgtin:4047111.012345.012345678901</parentID>

      <childEPCs>
            <epc>urn:epc:id:sgtin:4047111.000345.2323</epc>
      </childEPCs>

      <action>ADD</action>
      <bizStep>urn:epcglobal:cbv:bizstep:repairing</bizStep>
      <bizLocation>
            <id>urn:epc:id:sgln:4047111.12345.0001</id>
      </bizLocation>
</AggregationEvent>
```

Data 15: These two events denote the removal and addition (hence exchange/replacement) of a component inside an assembly. Concretely, events of this type are used to model the exchange of modules in KOS lamps.

There have no UIs been developed in the CIRC4Life Project so far, but EECC and Kosnic are working on detailed Requirements to support the demo. EECC will develop UIs to support the precise process as requirements are finished.

6.4 Impact Aggregation

The `Impact Aggregation Controller` offers endpoints to query for the aggregated impacts stored for an item by various identifiers. It can be tested through the access protected Swagger-UI at **https://circ4life.eecc.info/access/swagger-ui.html#/impact-aggregation-controller**.

This Service was developed to be the main data source for computing dynamic eco scores. In particular, data were to be delivered to NTU's automated ReCiPe computing service. Unfortunately, NTU has informed EECC that this service was cancelled. Nevertheless, the primary data on item specific impacts is available from the Traceability Module and EECC is looking into alternatives on how to compute dynamic product ratings e.g. using JS's Farm Carbon Calculator.

```
       [
         {
           "resourceList":
           {
             "epcsAndQuantityElementsAndResourceElements": [
               {
                 "value":
"https://circ4life.eecc.info/recycling/bin/obj/12345678901190514000011123 45"
               },
               {
                 "epcClass": "http://circ4life.eecc.info/material/copper",
                 "quantity": 0.5,
                 "uom": "KGM"
               }
             ]
           },
           "transportList":
           {
             "transportations": [
               {
                 "vehicle": "http://purl.org/vso/ns#Truck",
                 "distance":
                 {
                   "quantity": 10,
                   "uom": "KMT"
                 }
               }
```

```
              ]
            },
            "wasteList":
            {
              "epcsAndQuantityElementsAndWasteElements": [
                {
                  "waste": "http://circ4life.eecc.info/material/plastics",
                  "sink": "http://circ4life.eecc.info/sink/landfill",
                  "amount":
                  {
                    "quantity": 0.5,
                    "uom": "KGM"
                  }
                }
              ]
            },
            "impactList":
            {
              "impacts": [
                {
                  "impactType": "http://circ4life.eecc.info/type/land_usage",
                  "measure":
                  {
                    "quantity": 1,
                    "uom": "KMK"
                  }
                }
              ]
            }
          }
        ]
```

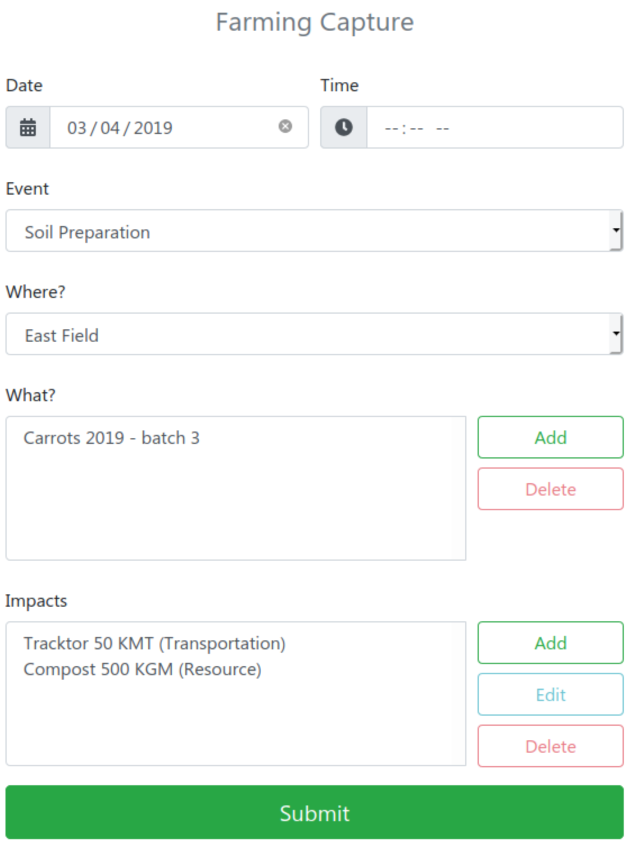
Data 16: Example for the aggregated impacts as returned by the **`Impact Aggregation Controller`**.

After for example posting the impact Data 8 for some item, querying the total impacts of this item yields the answer shown in Data 12.

EECC is offering to transform the output data into a more suitable format for the tool to use it, as soon as any tool is available for integration.

6.5 Impact Aggregation from Farming System Test

A system test for capturing impacts through various APIs, e.g. the farming UI (Figure 8, Figure 9, Figure 10) has been performed with subsequent data retrieval. See Data 17 for the aggregated impacts for carrots. This is meant as the input for the online LCA calculation, which is unfortunately not being developed by NTU as planned. EECC is working with JS to evaluate whether the Farm Carbon Calculator can be used as an online assessment tool.


Farming Capture

Date: 03 / 04 / 2019

Time: -- : -- --

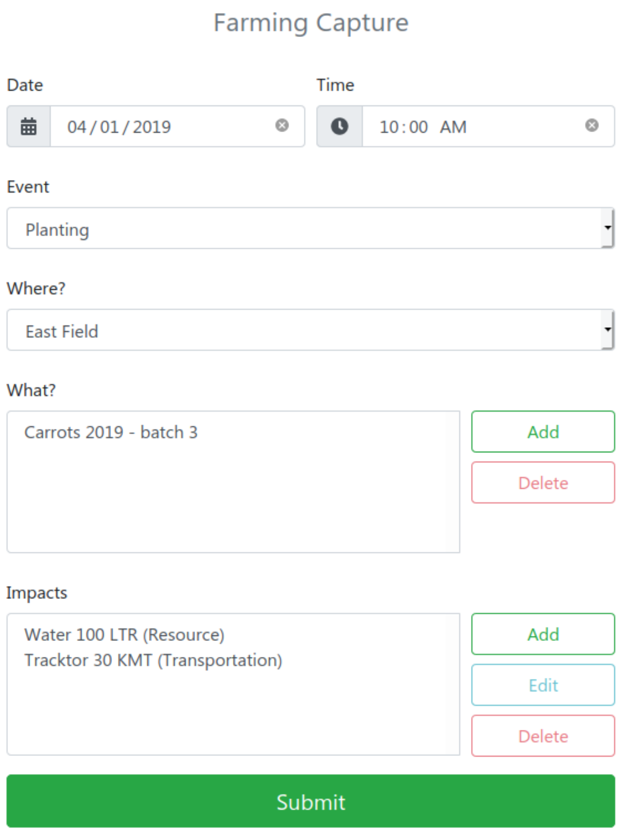
Event: Soil Preparation

Where?: East Field

What?: Carrots 2019 - batch 3

Impacts:
Tracktor 50 KMT (Transportation)
Compost 500 KGM (Resource)

Figure 8: Capturing impact for soil preparation for carrots.


Farming Capture

Date: 04 / 01 / 2019
Time: 10 : 00 AM

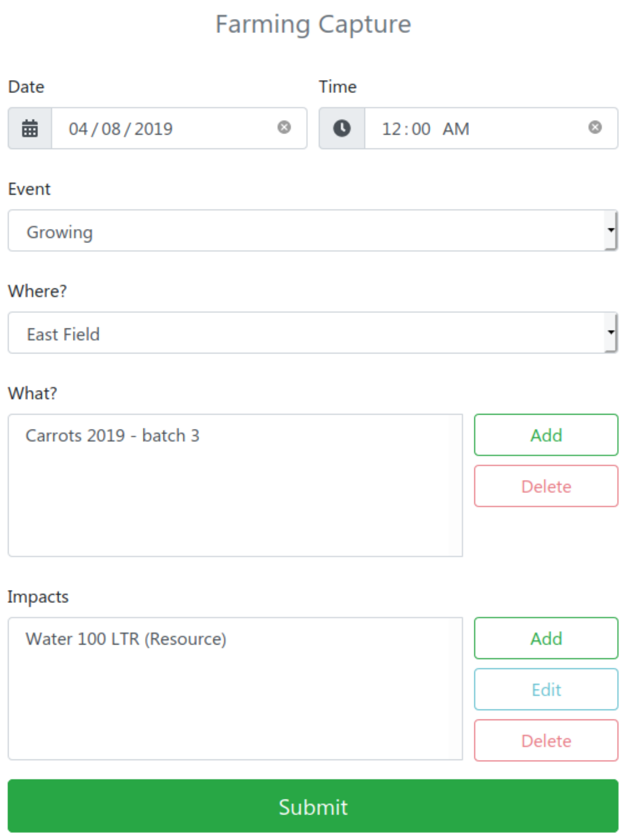
Event: Planting

Where?: East Field

What?: Carrots 2019 - batch 3

Impacts:
Water 100 LTR (Resource)
Tracktor 30 KMT (Transportation)

Figure 9: Capturing impact for planting/seeding the carrots


Farming Capture

Date: 04 / 08 / 2019

Time: 12 : 00 AM

Event: Growing

Where?: East Field

What?: Carrots 2019 - batch 3

Impacts: Water 100 LTR (Resource)

Figure 10: Capturing impact for carrots wile growing.

```
[
  {
    "Resourcelist":
    {
      "epcsAndQuantityElementsAndResourceElements": [
        {
          "resource": "Compost",
          "source": null,
          "amount":
          {
            "quantity": 500,
            "uom": "KGM"
          }
        }
      ]
    },
    "TransportList":
    {
      "transportations": [
        {
          "vehicle": "Tracktor",
          "distance":
          {
            "quantity": 100,
            "uom": "KMT"
          }
        }
      ]
    }
  },
  {
    "Resourcelist":
    {
      "epcsAndQuantityElementsAndResourceElements": [
        {
          "resource": "Water",
          "source": null,
          "amount":
          {
            "quantity": 100,
            "uom": "LTR"
          }
        }
      ]
    },
    "TransportList":
    {
      "transportations": [
        {
          "vehicle": "Tracktor",
          "distance":
          {
            "quantity": 30,
            "uom": "KMT"
          }
        }
      ]
    }
  },
  {
    "Resourcelist":
    {
      "epcsAndQuantityElementsAndResourceElements": [
        {
          "resource": "Water",
          "source": null,
          "amount":
          {
            "quantity": 100,
            "uom": "LTR"
          }
        }
      ]
    }
  }
]
```

Data 17: Example for the aggregated impacts as returned by the Impact Aggregation Controller[*]

[*] Aggregation impacts from the three farming events captured in Figure 8, Figure 9 and Figure 10

7 Conclusion

The Task 5.4 of WP 5 was dedicated to validate the correct functionality of the tools developed.

We have developed automated testing capabilities to test the software artefacts directly after each development iteration. This ensures that changes will not affect already developed parts of the system. The same holds for the deployment of the software to the servers each deployment is automatically checked and in case of failure rolled back to the version before if it is unsuccessful. This insures an always working system.

On an Interoperability level with ICT platform manual tests have been executed. These have been successful and we expect the system to work as designed.

In the section 6 we describe how the developed services may be used in the demos of CIRC4life. We executed tests with realistic datasets and can show that for example impacts collected throughout a product's lifecycle can be accessed by or been delivered to other systems. With this we see the Milestone 4 of CIRC4Life been achieved.

References

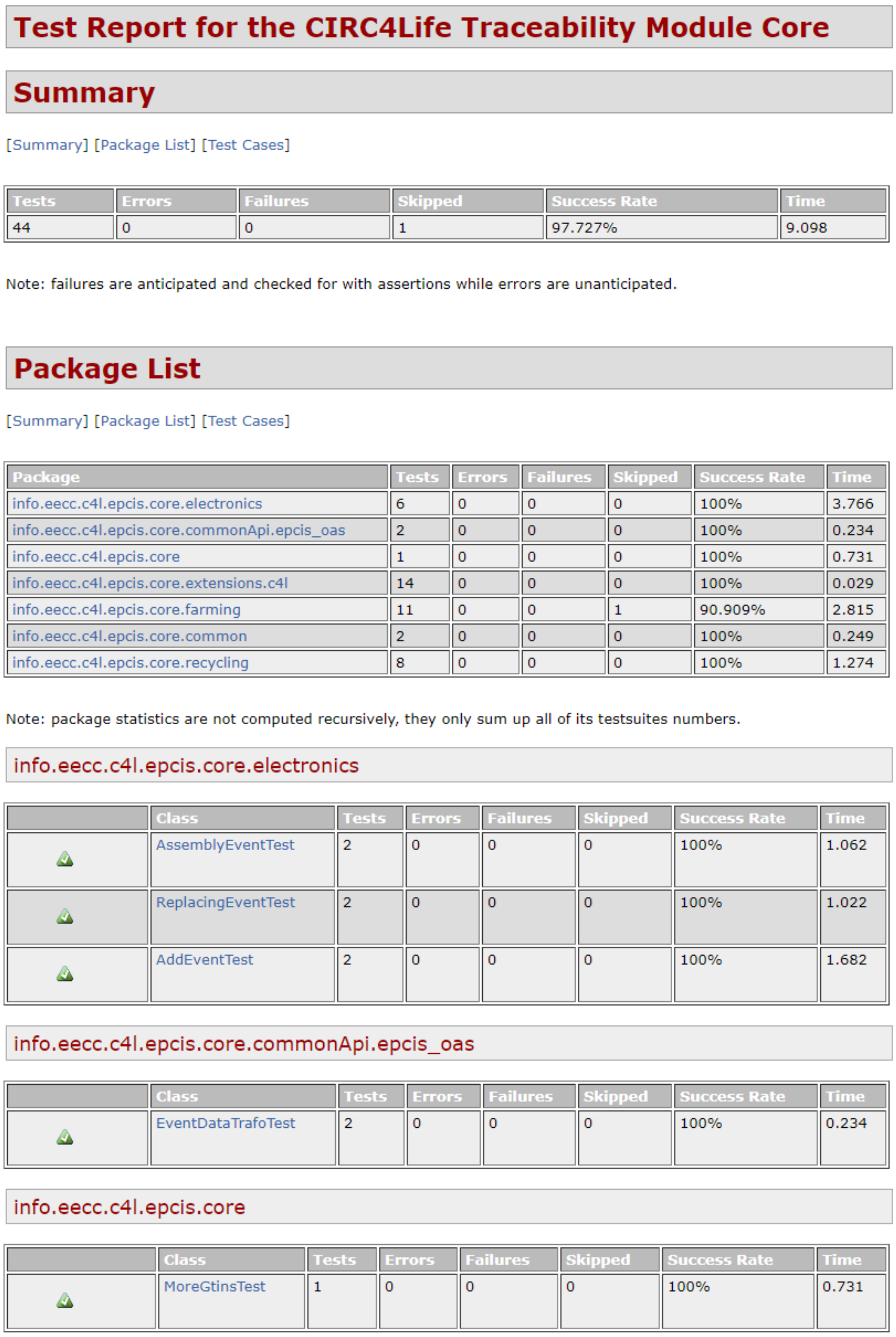
- Apache Software Foundation, 2019. *The Maven Project.* [Online]
  Available at: https://maven.apache.org/
  [Accessed 18 12 2019].
- Certbot, 2019. *Certbot.* [Online]
  Available at: https://certbot.eff.org/
  [Accessed 19 12 2019].
- Docker, 2019. *Docker Enterprise Container Platform.* [Online]
  Available at: https://www.docker.com/
  [Accessed 19 12 2019].
- EECC, 2019. *CIRC4Life impacts extension to the EPCIS standard - technical specification..* [Online]
  Available at: https://circ4life.eecc.info/doc/c4l-epcis-extensions.xsd .
  [Accessed 12 2019].
- EECC, 2019. *D 5.1: System specification and requirements for the traceability solutions, including access model,* s.l.: CIRC4Life.
- EECC, 2019. *D 5.2: Development report and documentation for traceability components and tools,* s.l.: CIRC4Life.
- EECC, 2019. *D 5.3: Report on the implementation of Traceability Solutions in three CEBMs and Interface development for ICT platform integration,* s.l.: CIRC4Life.
- European Union, 2018. *EUR-Lex.* [Online]
  Available at: https://eur-lex.europa.eu
- GS1 Global, 2016. *EPC Information Services (EPCIS) Specification - Release 1.2,* Brussels: Global Standards One.
- GS1 Global, 2017. *Core Business Vocabulary (CBV)Specification - Release 1.2.2,* Brussels: Global Standards One.
- ICCS, 2019. *D 4.1: System Specification including traceability matrix –functional components vs.requirements,* s.l.: CIRC4Life.
- ICCS, 2019. *D 4.2: Report on Information Logistics Systems development and resulting systems and processes,* s.l.: CIRC4Life.
- JUnit, 219. *JUnit 5.* [Online]
  Available at: https://junit.org/junit5/
  [Accessed 29 12 2019].
- Let's encrypt, 2019. *Let's encrypt - Free SSL/TLS Certificates.* [Online]
  Available at: https://letsencrypt.org/
  [Accessed 19 12 2019].
- OAuth, 2019. *OAuth 2.0.* [Online]
  Available at: https://oauth.net/2/
- OpenID, 2019. *OpenID Connect.* [Online]
  Available at: https://openid.net/connect/

RedHat, 2019. *KeyCloak.* [Online]
Available at: https://www.keycloak.org/ .

8 Appendix

8.1 Unit and Internal Integration Test Reports

8.1.1 Traceability Module Core Test Report



Test Report for the CIRC4Life Traceability Module Core

Summary

[Summary] [Package List] [Test Cases]

Tests	Errors	Failures	Skipped	Success Rate	Time
44	0	0	1	97.727%	9.098

Note: failures are anticipated and checked for with assertions while errors are unanticipated.

Package List

[Summary] [Package List] [Test Cases]

Package	Tests	Errors	Failures	Skipped	Success Rate	Time
info.eecc.c4l.epcis.core.electronics	6	0	0	0	100%	3.766
info.eecc.c4l.epcis.core.commonApi.epcis_oas	2	0	0	0	100%	0.234
info.eecc.c4l.epcis.core	1	0	0	0	100%	0.731
info.eecc.c4l.epcis.core.extensions.c4l	14	0	0	0	100%	0.029
info.eecc.c4l.epcis.core.farming	11	0	0	1	90.909%	2.815
info.eecc.c4l.epcis.core.common	2	0	0	0	100%	0.249
info.eecc.c4l.epcis.core.recycling	8	0	0	0	100%	1.274

Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.

info.eecc.c4l.epcis.core.electronics

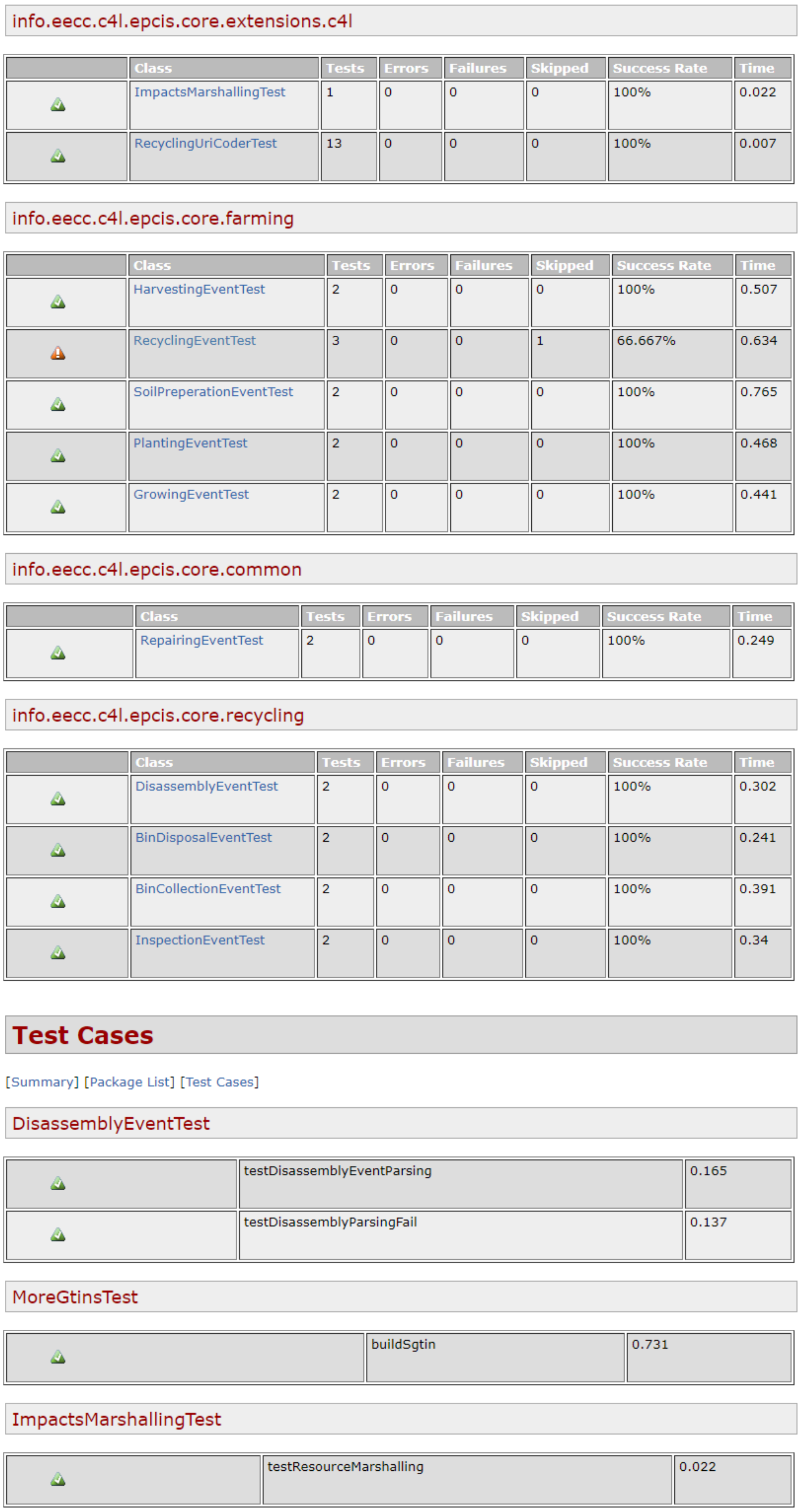
	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	AssemblyEventTest	2	0	0	0	100%	1.062
✓	ReplacingEventTest	2	0	0	0	100%	1.022
✓	AddEventTest	2	0	0	0	100%	1.682

info.eecc.c4l.epcis.core.commonApi.epcis_oas

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	EventDataTrafoTest	2	0	0	0	100%	0.234

info.eecc.c4l.epcis.core

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	MoreGtinsTest	1	0	0	0	100%	0.731



info.eecc.c4l.epcis.core.extensions.c4l

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	ImpactsMarshallingTest	1	0	0	0	100%	0.022
✓	RecyclingUriCoderTest	13	0	0	0	100%	0.007

info.eecc.c4l.epcis.core.farming

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	HarvestingEventTest	2	0	0	0	100%	0.507
⚠	RecyclingEventTest	3	0	0	1	66.667%	0.634
✓	SoilPreperationEventTest	2	0	0	0	100%	0.765
✓	PlantingEventTest	2	0	0	0	100%	0.468
✓	GrowingEventTest	2	0	0	0	100%	0.441

info.eecc.c4l.epcis.core.common

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	RepairingEventTest	2	0	0	0	100%	0.249

info.eecc.c4l.epcis.core.recycling

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	DisassemblyEventTest	2	0	0	0	100%	0.302
✓	BinDisposalEventTest	2	0	0	0	100%	0.241
✓	BinCollectionEventTest	2	0	0	0	100%	0.391
✓	InspectionEventTest	2	0	0	0	100%	0.34

Test Cases

[Summary] [Package List] [Test Cases]

DisassemblyEventTest

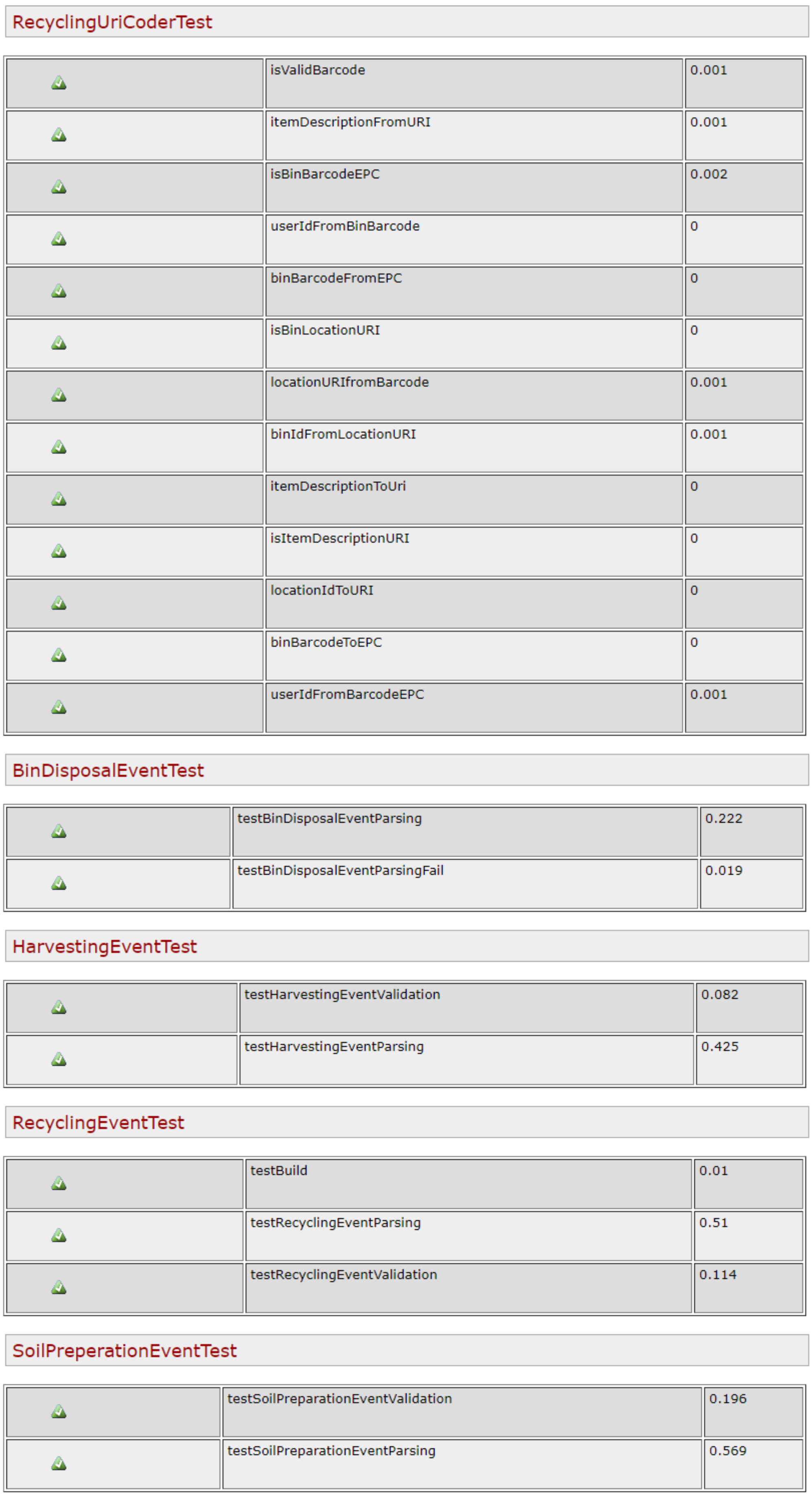
✓	testDisassemblyEventParsing	0.165
✓	testDisassemblyParsingFail	0.137

MoreGtinsTest

✓	buildSgtin	0.731

ImpactsMarshallingTest

✓	testResourceMarshalling	0.022



RecyclingUriCoderTest

✓	isValidBarcode	0.001
✓	itemDescriptionFromURI	0.001
✓	isBinBarcodeEPC	0.002
✓	userIdFromBinBarcode	0
✓	binBarcodeFromEPC	0
✓	isBinLocationURI	0
✓	locationURIfromBarcode	0.001
✓	binIdFromLocationURI	0.001
✓	itemDescriptionToUri	0
✓	isItemDescriptionURI	0
✓	locationIdToURI	0
✓	binBarcodeToEPC	0
✓	userIdFromBarcodeEPC	0.001

BinDisposalEventTest

✓	testBinDisposalEventParsing	0.222
✓	testBinDisposalEventParsingFail	0.019

HarvestingEventTest

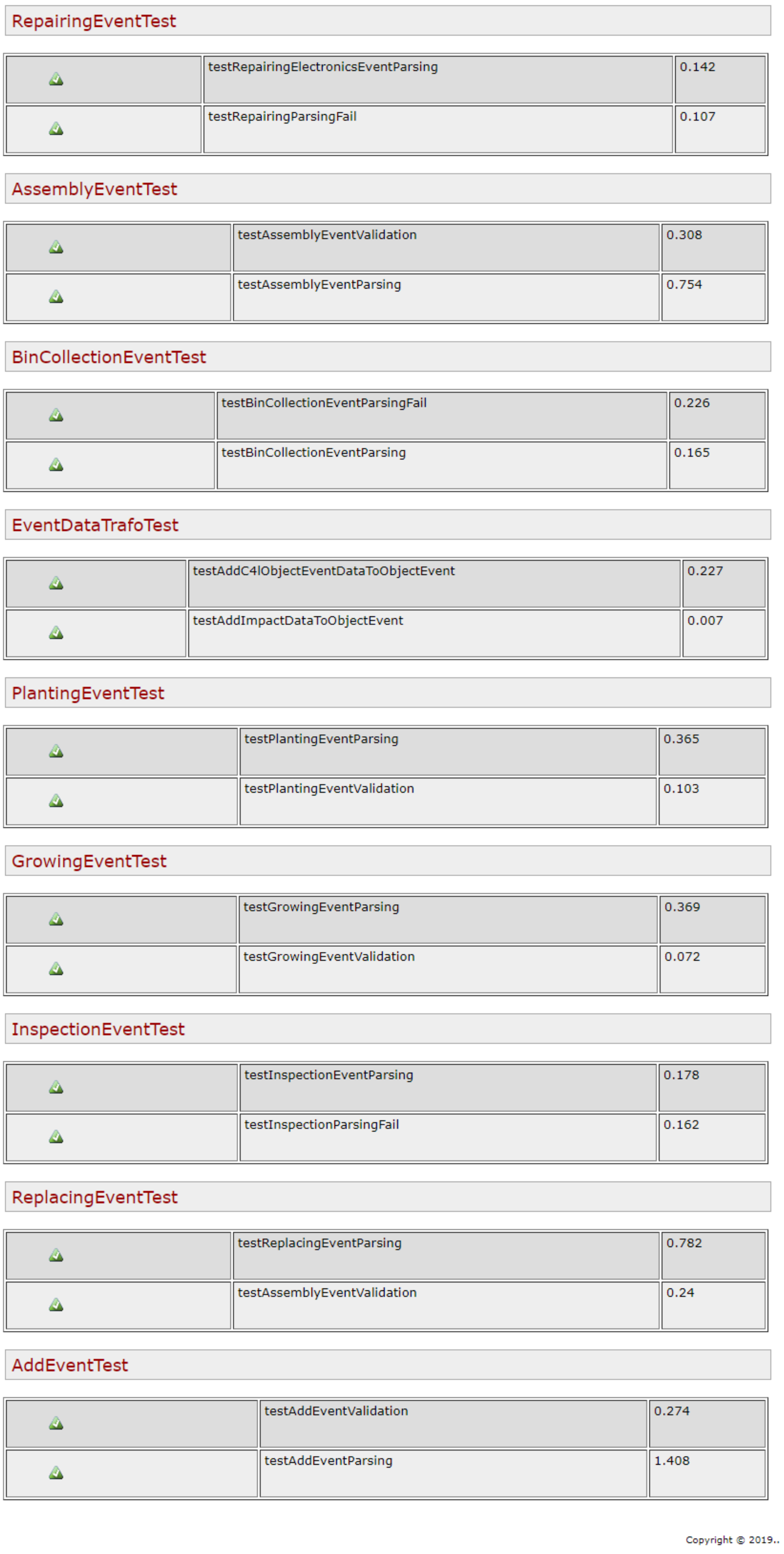
✓	testHarvestingEventValidation	0.082
✓	testHarvestingEventParsing	0.425

RecyclingEventTest

✓	testBuild	0.01
✓	testRecyclingEventParsing	0.51
✓	testRecyclingEventValidation	0.114

SoilPreperationEventTest

✓	testSoilPreparationEventValidation	0.196
✓	testSoilPreparationEventParsing	0.569



RepairingEventTest

✓	testRepairingElectronicsEventParsing	0.142
✓	testRepairingParsingFail	0.107

AssemblyEventTest

✓	testAssemblyEventValidation	0.308
✓	testAssemblyEventParsing	0.754

BinCollectionEventTest

✓	testBinCollectionEventParsingFail	0.226
✓	testBinCollectionEventParsing	0.165

EventDataTrafoTest

✓	testAddC4lObjectEventDataToObjectEvent	0.227
✓	testAddImpactDataToObjectEvent	0.007

PlantingEventTest

✓	testPlantingEventParsing	0.365
✓	testPlantingEventValidation	0.103

GrowingEventTest

✓	testGrowingEventParsing	0.369
✓	testGrowingEventValidation	0.072

InspectionEventTest

✓	testInspectionEventParsing	0.178
✓	testInspectionParsingFail	0.162

ReplacingEventTest

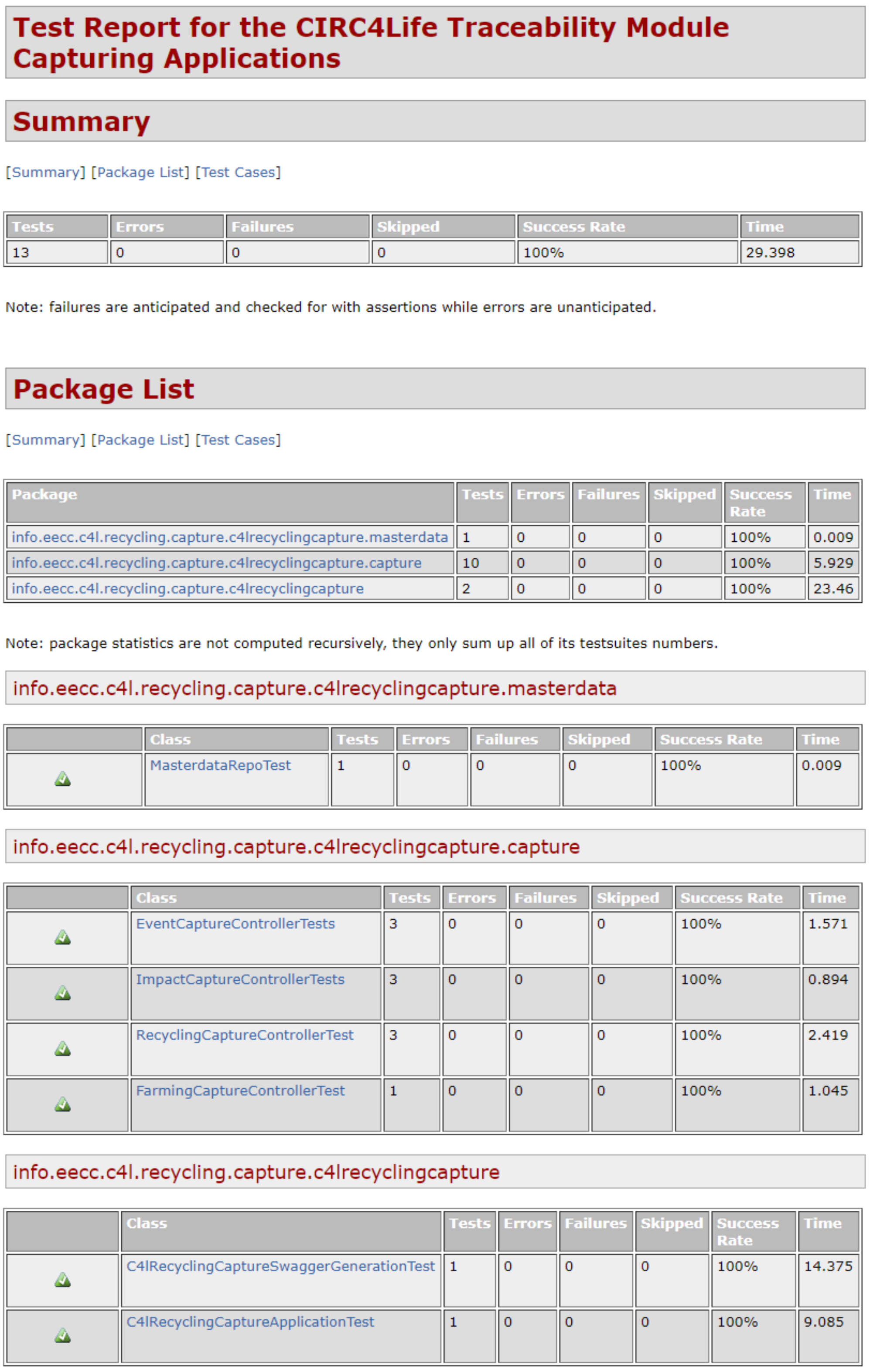
✓	testReplacingEventParsing	0.782
✓	testAssemblyEventValidation	0.24

AddEventTest

✓	testAddEventValidation	0.274
✓	testAddEventParsing	1.408

Copyright © 2019..

8.1.2 Capturing Applications Test Report



Test Report for the CIRC4Life Traceability Module Capturing Applications

Summary

[Summary] [Package List] [Test Cases]

Tests	Errors	Failures	Skipped	Success Rate	Time
13	0	0	0	100%	29.398

Note: failures are anticipated and checked for with assertions while errors are unanticipated.

Package List

[Summary] [Package List] [Test Cases]

Package	Tests	Errors	Failures	Skipped	Success Rate	Time
info.eecc.c4l.recycling.capture.c4lrecyclingcapture.masterdata	1	0	0	0	100%	0.009
info.eecc.c4l.recycling.capture.c4lrecyclingcapture.capture	10	0	0	0	100%	5.929
info.eecc.c4l.recycling.capture.c4lrecyclingcapture	2	0	0	0	100%	23.46

Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.

info.eecc.c4l.recycling.capture.c4lrecyclingcapture.masterdata

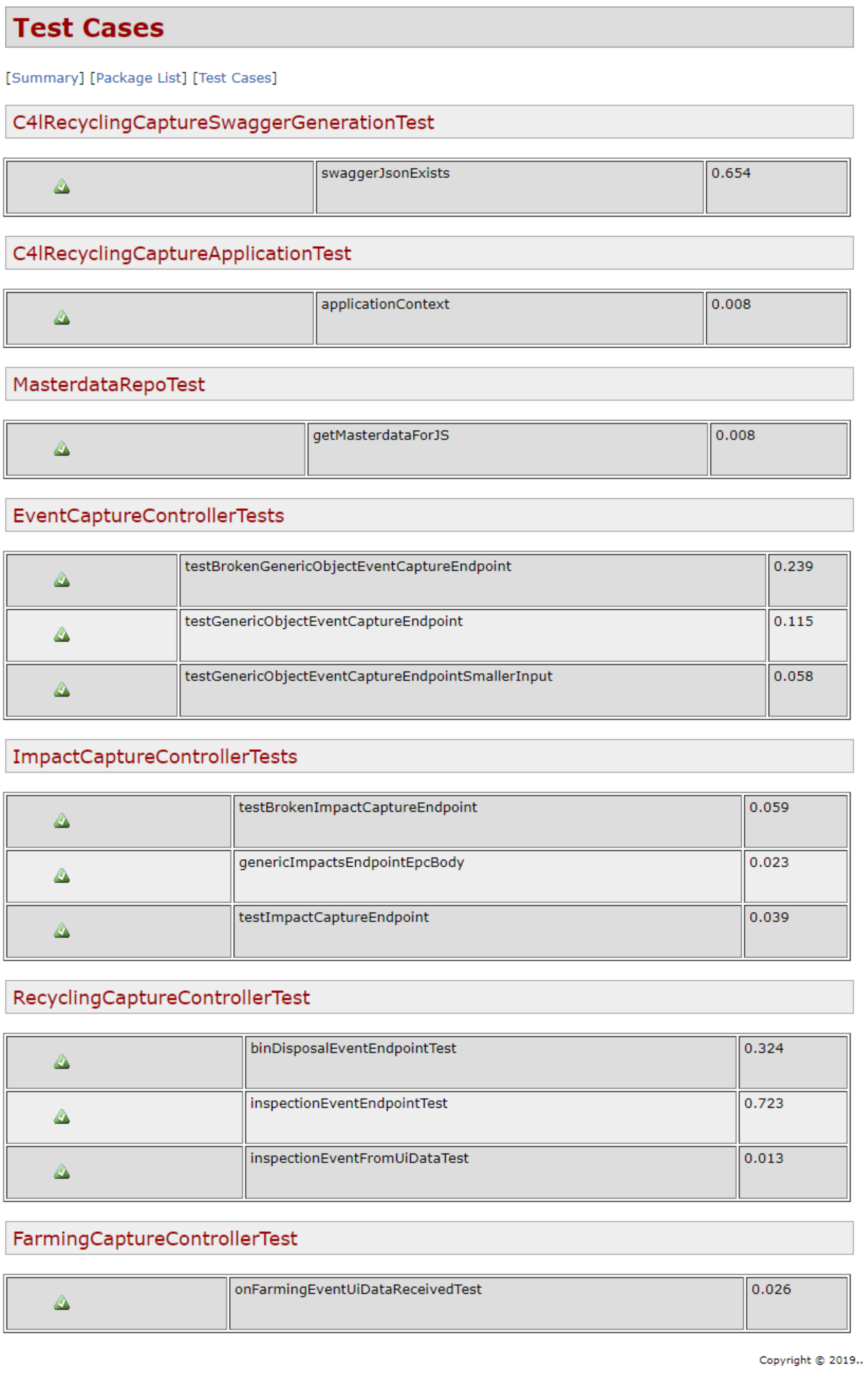
	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	MasterdataRepoTest	1	0	0	0	100%	0.009

info.eecc.c4l.recycling.capture.c4lrecyclingcapture.capture

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	EventCaptureControllerTests	3	0	0	0	100%	1.571
✓	ImpactCaptureControllerTests	3	0	0	0	100%	0.894
✓	RecyclingCaptureControllerTest	3	0	0	0	100%	2.419
✓	FarmingCaptureControllerTest	1	0	0	0	100%	1.045

info.eecc.c4l.recycling.capture.c4lrecyclingcapture

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	C4lRecyclingCaptureSwaggerGenerationTest	1	0	0	0	100%	14.375
✓	C4lRecyclingCaptureApplicationTest	1	0	0	0	100%	9.085



Test Cases

[Summary] [Package List] [Test Cases]

C4lRecyclingCaptureSwaggerGenerationTest

✓	swaggerJsonExists	0.654

C4lRecyclingCaptureApplicationTest

✓	applicationContext	0.008

MasterdataRepoTest

✓	getMasterdataForJS	0.008

EventCaptureControllerTests

✓	testBrokenGenericObjectEventCaptureEndpoint	0.239
✓	testGenericObjectEventCaptureEndpoint	0.115
✓	testGenericObjectEventCaptureEndpointSmallerInput	0.058

ImpactCaptureControllerTests

✓	testBrokenImpactCaptureEndpoint	0.059
✓	genericImpactsEndpointEpcBody	0.023
✓	testImpactCaptureEndpoint	0.039

RecyclingCaptureControllerTest

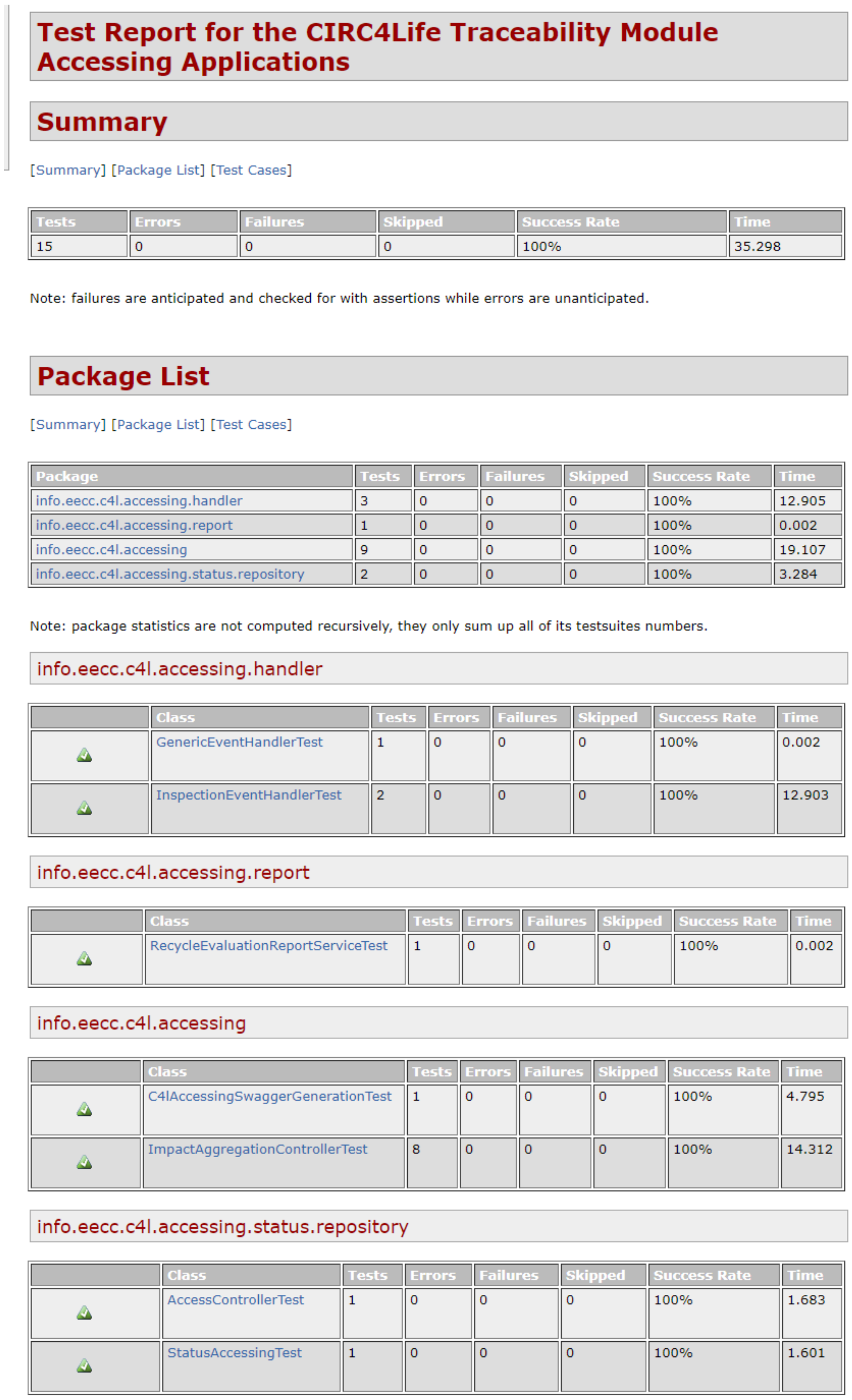
✓	binDisposalEventEndpointTest	0.324
✓	inspectionEventEndpointTest	0.723
✓	inspectionEventFromUiDataTest	0.013

FarmingCaptureControllerTest

✓	onFarmingEventUiDataReceivedTest	0.026

Copyright © 2019..

8.1.3 Accessing Applications Test Report



Test Report for the CIRC4Life Traceability Module Accessing Applications

Summary

[Summary] [Package List] [Test Cases]

Tests	Errors	Failures	Skipped	Success Rate	Time
15	0	0	0	100%	35.298

Note: failures are anticipated and checked for with assertions while errors are unanticipated.

Package List

[Summary] [Package List] [Test Cases]

Package	Tests	Errors	Failures	Skipped	Success Rate	Time
info.eecc.c4l.accessing.handler	3	0	0	0	100%	12.905
info.eecc.c4l.accessing.report	1	0	0	0	100%	0.002
info.eecc.c4l.accessing	9	0	0	0	100%	19.107
info.eecc.c4l.accessing.status.repository	2	0	0	0	100%	3.284

Note: package statistics are not computed recursively, they only sum up all of its testsuites numbers.

info.eecc.c4l.accessing.handler

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	GenericEventHandlerTest	1	0	0	0	100%	0.002
✓	InspectionEventHandlerTest	2	0	0	0	100%	12.903

info.eecc.c4l.accessing.report

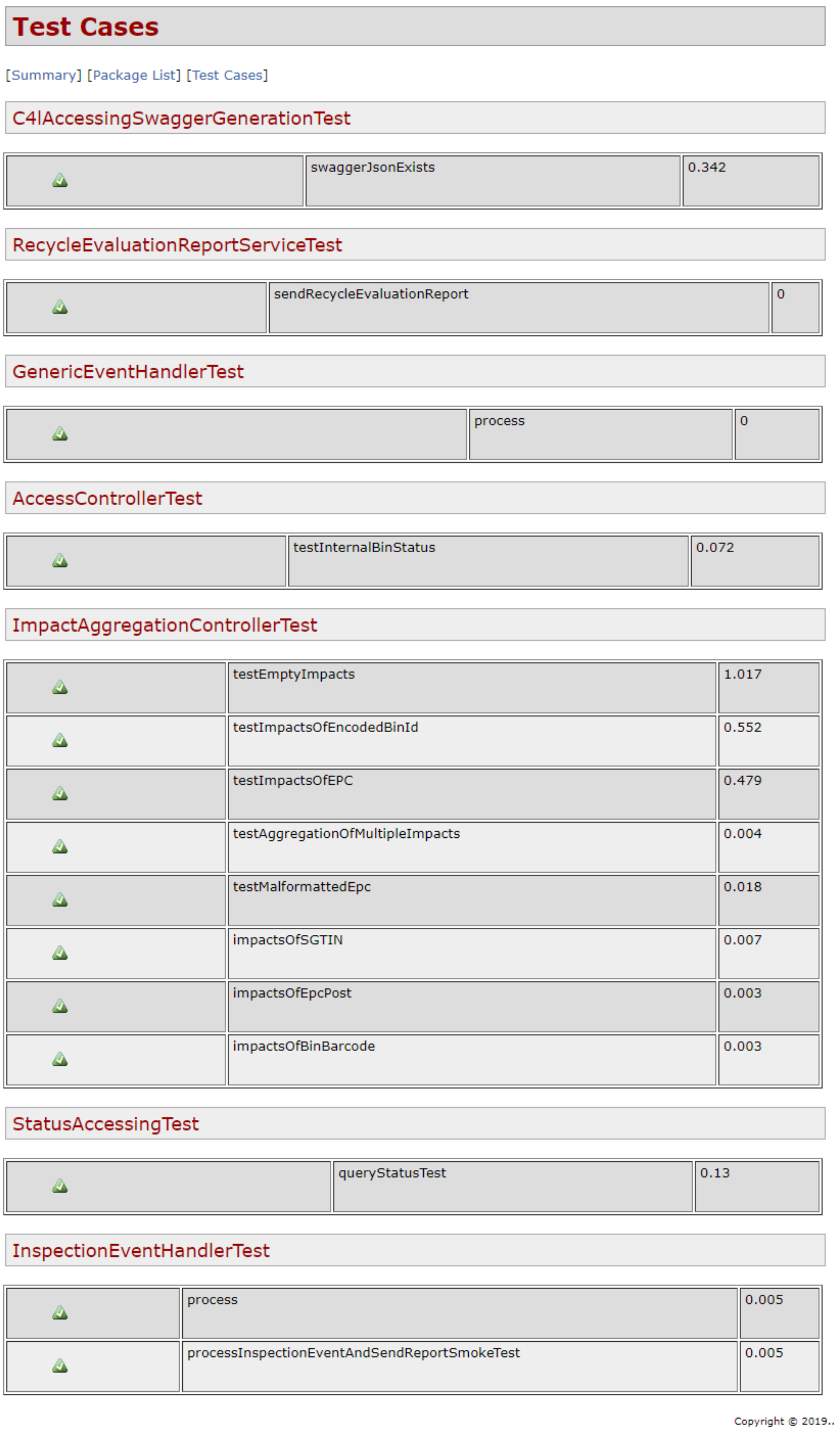
	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	RecycleEvaluationReportServiceTest	1	0	0	0	100%	0.002

info.eecc.c4l.accessing

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	C4lAccessingSwaggerGenerationTest	1	0	0	0	100%	4.795
✓	ImpactAggregationControllerTest	8	0	0	0	100%	14.312

info.eecc.c4l.accessing.status.repository

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
✓	AccessControllerTest	1	0	0	0	100%	1.683
✓	StatusAccessingTest	1	0	0	0	100%	1.601



Test Cases

[Summary] [Package List] [Test Cases]

C4lAccessingSwaggerGenerationTest

✓	swaggerJsonExists	0.342

RecycleEvaluationReportServiceTest

✓	sendRecycleEvaluationReport	0

GenericEventHandlerTest

✓	process	0

AccessControllerTest

✓	testInternalBinStatus	0.072

ImpactAggregationControllerTest

✓	testEmptyImpacts	1.017
✓	testImpactsOfEncodedBinId	0.552
✓	testImpactsOfEPC	0.479
✓	testAggregationOfMultipleImpacts	0.004
✓	testMalformattedEpc	0.018
✓	impactsOfSGTIN	0.007
✓	impactsOfEpcPost	0.003
✓	impactsOfBinBarcode	0.003

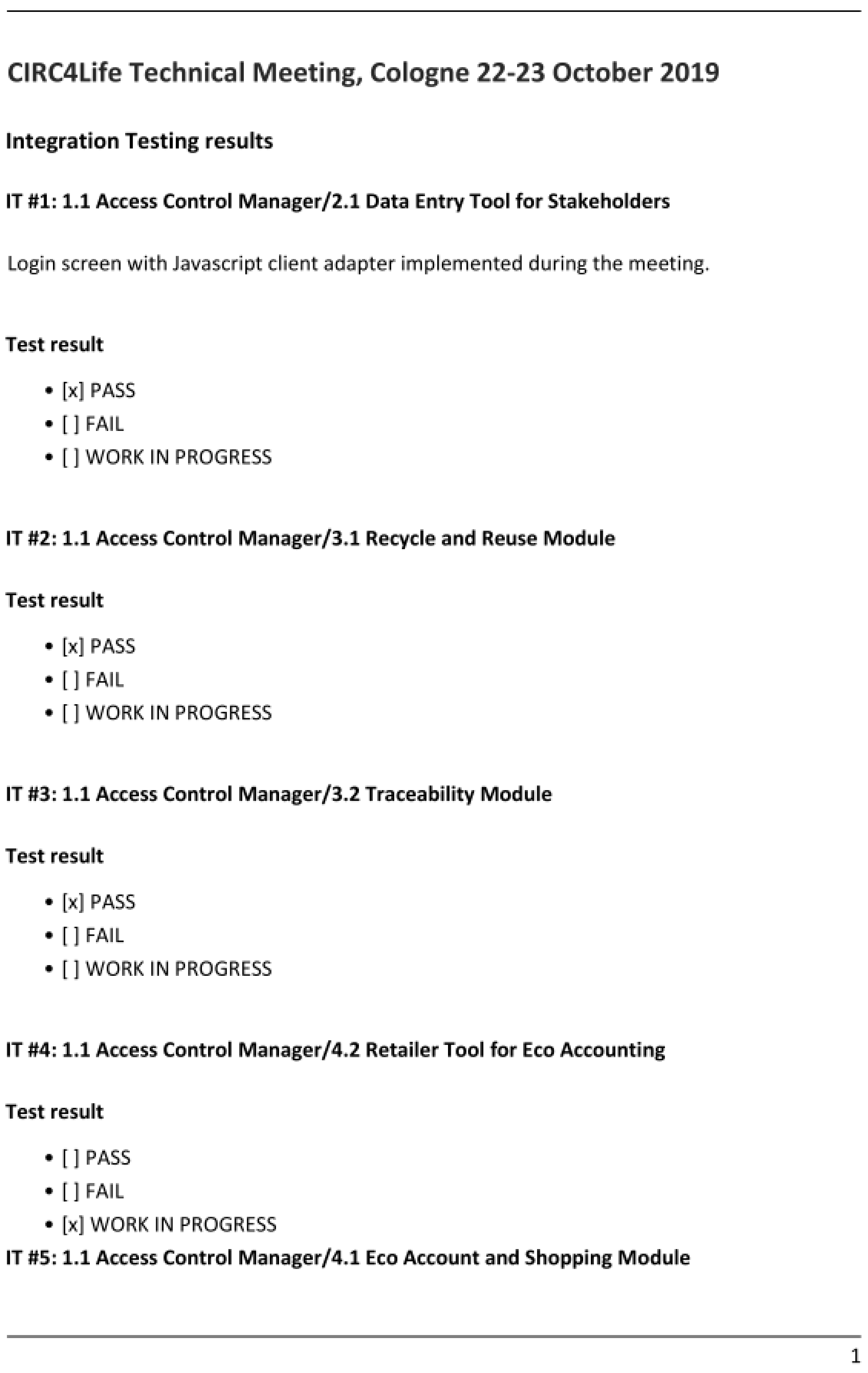
StatusAccessingTest

✓	queryStatusTest	0.13

InspectionEventHandlerTest

✓	process	0.005
✓	processInspectionEventAndSendReportSmokeTest	0.005

8.2 External Integration Test Protocols



CIRC4Life Technical Meeting, Cologne 22-23 October 2019

Integration Testing results

IT #1: 1.1 Access Control Manager/2.1 Data Entry Tool for Stakeholders

Login screen with Javascript client adapter implemented during the meeting.

Test result

- [x] PASS
- [ ] FAIL
- [ ] WORK IN PROGRESS

IT #2: 1.1 Access Control Manager/3.1 Recycle and Reuse Module

Test result

- [x] PASS
- [ ] FAIL
- [ ] WORK IN PROGRESS

IT #3: 1.1 Access Control Manager/3.2 Traceability Module

Test result

- [x] PASS
- [ ] FAIL
- [ ] WORK IN PROGRESS

IT #4: 1.1 Access Control Manager/4.2 Retailer Tool for Eco Accounting

Test result

- [ ] PASS
- [ ] FAIL
- [x] WORK IN PROGRESS

IT #5: 1.1 Access Control Manager/4.1 Eco Account and Shopping Module

1

Test result

- [x] PASS
- [ ] FAIL
- [ ] WORK IN PROGRESS

IT #6: 2.2 Intermediate Products Data Export Web Service/1.1.2 Data Gateway

Test result

- [ ] PASS
- [ ] FAIL
- [x] WORK IN PROGRESS

IT #7: 1.1.1 ICT Web Services/3.2 Traceability Module

Test result

- [x] PASS
- [ ] FAIL
- [ ] WORK IN PROGRESS

IT #8: 1.1.1 ICT Web Services/4.1 Eco Account and Shopping Module

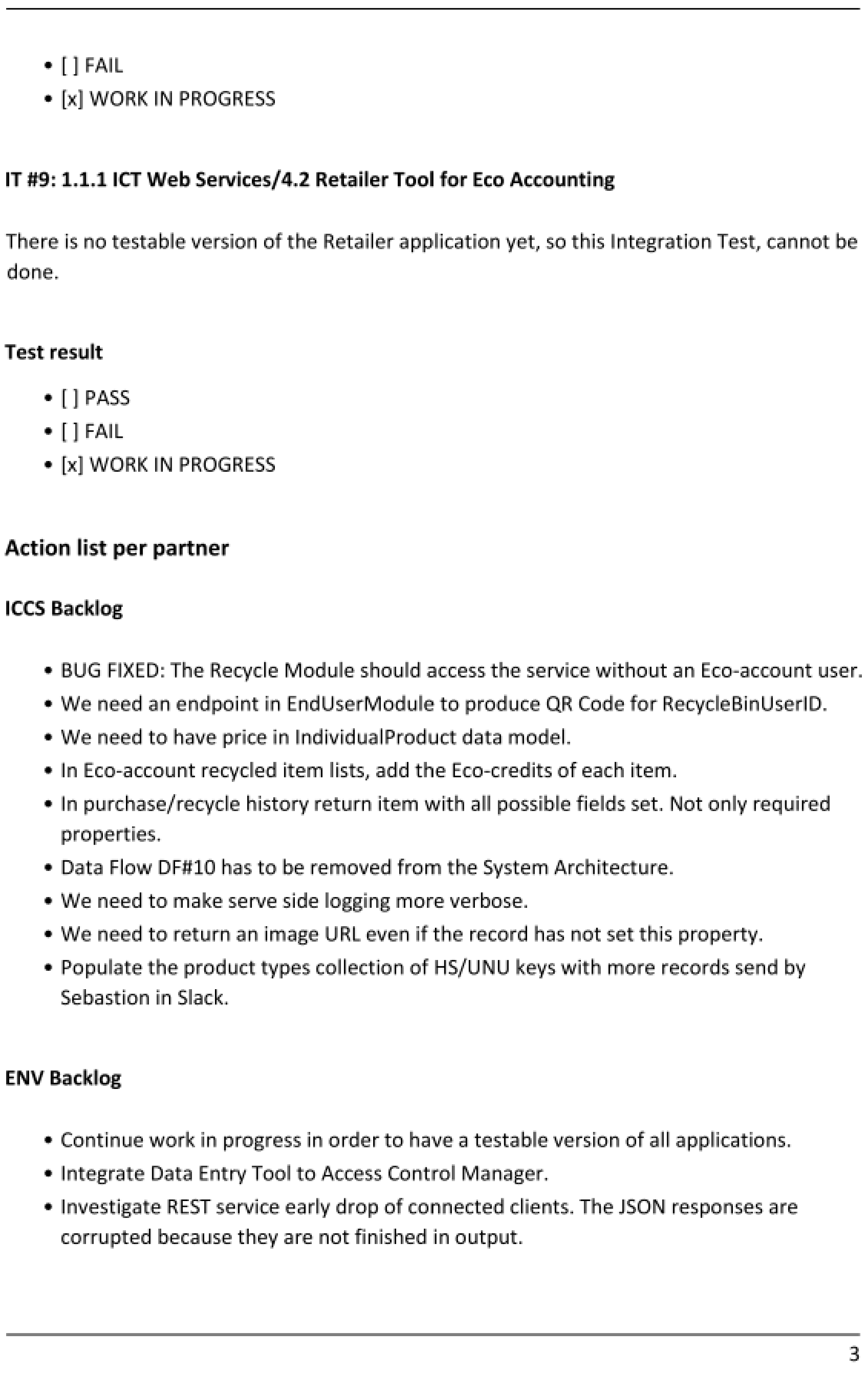
The mobile app has not yet fully integrated with all EndUserModule endpoints. The ones that are integrated and tested are the following:

- https://circ4life.iccs.gr/EndUserModule/resources/ecoaccount/ecobalance
- https://circ4life.iccs.gr/EndUserModule/resources/ecoaccount/user_record
- https://circ4life.iccs.gr/EndUserModule/resources/ecoaccount/history
- https://circ4life.iccs.gr/EndUserModule/resources/ecoaccount/login
- https://circ4life.iccs.gr/EndUserModule/resources/ecoaccount/logout
- https://circ4life.iccs.gr/EndUserModule/resources/ecoshopping/product/info/04047111123453

TheactionlistforICCSandNTUforthistestislargeenoughtoconsideritasWorkinprogress.

Test result

- [ ] PASS



- [ ] FAIL
- [x] WORK IN PROGRESS

IT #9: 1.1.1 ICT Web Services/4.2 Retailer Tool for Eco Accounting

There is no testable version of the Retailer application yet, so this Integration Test, cannot be done.

Test result

- [ ] PASS
- [ ] FAIL
- [x] WORK IN PROGRESS

Action list per partner

ICCS Backlog

- BUG FIXED: The Recycle Module should access the service without an Eco-account user.
- We need an endpoint in EndUserModule to produce QR Code for RecycleBinUserID.
- We need to have price in IndividualProduct data model.
- In Eco-account recycled item lists, add the Eco-credits of each item.
- In purchase/recycle history return item with all possible fields set. Not only required properties.
- Data Flow DF#10 has to be removed from the System Architecture.
- We need to make serve side logging more verbose.
- We need to return an image URL even if the record has not set this property.
- Populate the product types collection of HS/UNU keys with more records send by Sebastion in Slack.

ENV Backlog

- Continue work in progress in order to have a testable version of all applications.
- Integrate Data Entry Tool to Access Control Manager.
- Investigate REST service early drop of connected clients. The JSON responses are corrupted because they are not finished in output.

EECC Backlog

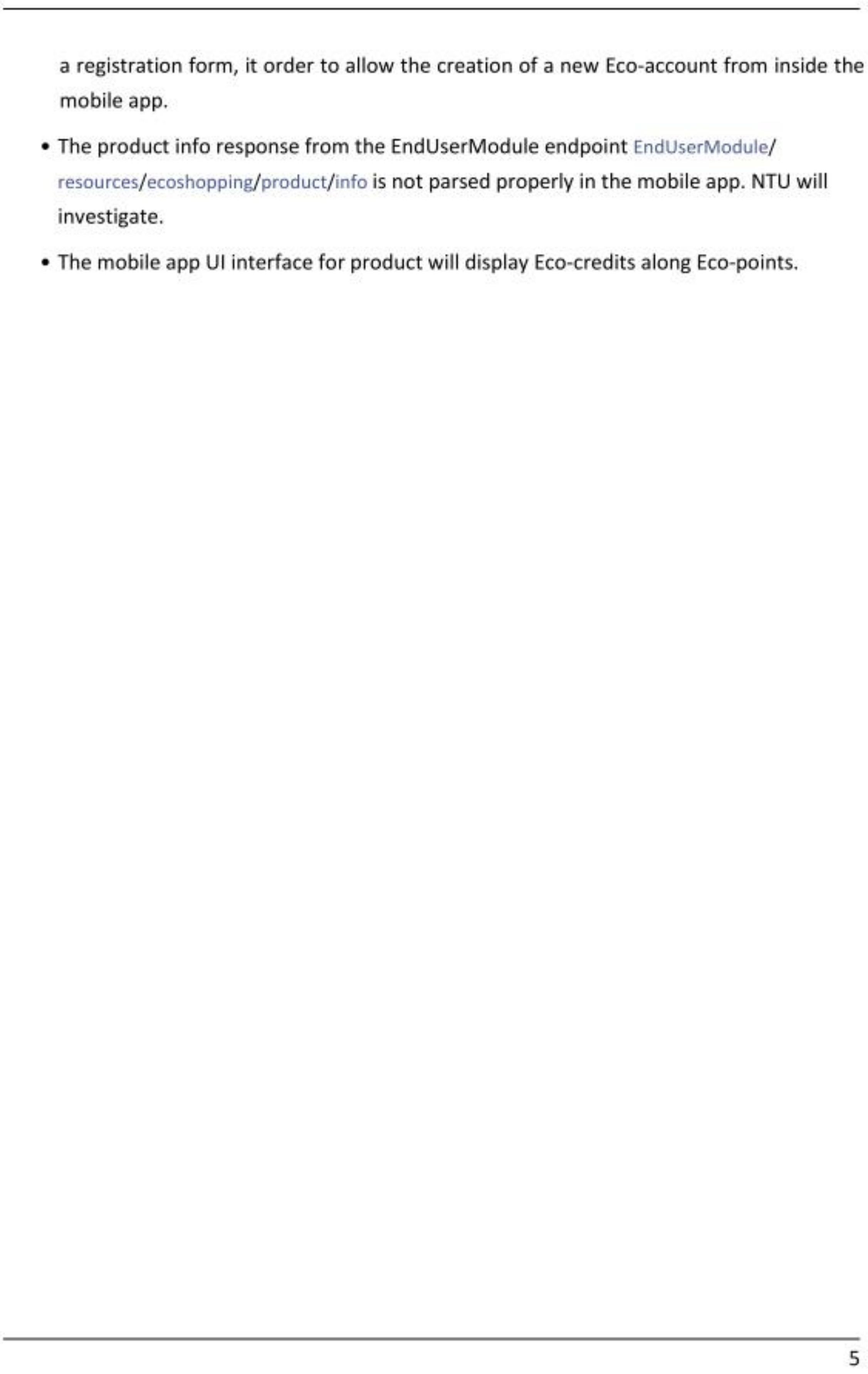
- IntegrateproducttypesUIwiththeICTPlatformendpointRecycleModule/resources /product_types

NTU Backlog

- Continue work in progress in order to have a testable version of all applications.
- Define data model to for Sustainability -> MasterData.LifeCycle.??? and Recycle Scheme
-> MasterData.LifeCycle.Resources.Recyclability.Scheme -> { a: "" , b: "" }. These properties are present in the Master Product record and apply to items representing the same product.
- In mobile app when a property for a product is missing from the ICT platform response or database (e.g. image) the app should display a placeholder item that showcases that this property is not available.
- Study the OpenAPI spec user/system requirements and get in contact with end users (demo owners/users) in order to find out which API functionality is to be displayed in mobileapplicationorotherend-userapps. Exampletheendpointhttps://circ4life .iccs.gr/EndUserModule/resources/ecoshopping/product/manufacturer is not used by the mobile app right now but it has been derived from user requirements USER_U_03, USER_U_04, USER_U_06.
- To be implemented in the mobile app:

```
<https://circ4life.iccs.gr/EndUserModule/resources/
    ecoaccount/anonymous_consumerid>
<https://circ4life.iccs.gr/EndUserModule/resources/
    ecoaccount/anonymous_consumerid/qrcode>
<https://circ4life.iccs.gr/EndUserModule/resources/ ecoaccount/register>
```

The first 2 are required for anonymous Eco-account access from the RetailerModule. The scenario is the following: the user logins to his Eco-account with the mobile app. When he visits a retailer store that supports the system (RetailerModule integration), he can present an anonymous ID of his Eco-account via Text or QR Code. This string or QR code image is returned by the first two endpoints above. Then the retailer can update the Eco-account purchase list using the anonymous ID. The last one should be integrated along



a registration form, it order to allow the creation of a new Eco-account from inside the mobile app.

- The product info response from the EndUserModule endpoint EndUserModule/resources/ecoshopping/product/info is not parsed properly in the mobile app. NTU will investigate.
- The mobile app UI interface for product will display Eco-credits along Eco-points.

5