InventiveTr@ining – Inven!RA architecture Activity Provider modules for online tracking of microelectronics student projects

Duarte Cota[1], Tiago Cruzeiro[2], Dennis Beck[3], António Coelho[2, 4], Leonel Morgado[1, 4]

[1] Universidade Aberta, Portugal, [2] Universidade do Porto, Portugal, [3] University of Arkansas, USA, [4] INESC TEC, Portugal
duartenunocota@gmail.com, tiago.c.1991@gmail.com, debeck@uark.edu, acoelho@fe.up.pt, Leonel.Morgado@uab.pt

Abstract: The Inven!RA architecture is an approach for online tracking of progression towards learning objectives, from analytics of distributed learning activities, provided by multiple third parties. However, there are few examples on how to implement such third-party learning activities, known as Activity Provider modules. We followed the Inven!RA architecture interfacing specification to create and implement two sample learning activities: a technical documentation analysis activity and an Arduino microelectronics programming activity. Integration tests with an Inven!RA architecture prototype confirmed the adequacy of this implementation. Thus, these samples provide clarification on how to design and develop Inven!RA Activity Provider modules.

Keywords: Inven!RA, learning analytics, learning management systems, Arduino, microelectronics learning

Título: InventiveTr@ining – Módulos Activity Provider da arquitetura Inven!RA para acompanhamento online de projetos de alunos de microeletrónica

Resumo: A arquitetura Inven!RA é uma abordagem para o acompanhamento online da evolução face a objetivos de aprendizagem, através de dados analíticos de atividades de aprendizagem distribuídas, proporcionadas por um leque variado de entidades externas. Como são escassos os exemplos de implementação destas atividades de aprendizagem externas, designadas por módulos de Prestadores de Atividades**, s**eguimos a especificação de interfaces da arquitetura para criar e implementar dois exemplos de atividades de aprendizagem: uma atividade de análise de documentação técnica e uma de programação de microeletrónica com Arduino. Testes de integração com um protótipo da Inven!RA confirmaram a adequação destas implementações. Consequentemente, proporcionam clarificação quanto à forma de conceber e desenvolver módulos de Prestadores de Atividades para a arquitetura Inven!RA.

Palavras-chave: Inven!RA, análise da aprendizagem, sistemas de gestão da aprendizagem, Arduino, aprendizagem de microeletrónica

1. Introduction

The heavy workload associated with teacher use of rich and diversified learning activities hampers their widespread adoption in the field of education. This workload derives from the diversity of student activity information that teachers need to be aware of in the classroom, as well as the variety of information they need to consider when deciding upon feedback or assessment. Game-based learning is one example of an advanced pedagogical activity that suffers from this very problem (Marklund & Taylor, 2016). The BEACONING project designed and implemented a broker pattern architecture approach that tackled this problem by merging a game narrative with third-party educational minigame activities associated with learning objectives (Bourazeri et al., 2017). This approach was then leveraged by the Inven!RA architecture (pronounced [ɪn-vehn-i-rɑ]) to enable third-party learning activity orchestration and tracking without being tied to a particular game narrative or without restriction to game-based learning (Cruzeiro, 2020). Its inception presented limited information on third-party activities for orchestration, specifically for a serious pervasive games frontend (Coelho et al., 2020). The name represents "a medium for Inventive agency amidst Reticular ecosystems of Atopic habitats, within which knowledge emerges" (Cruzeiro, 2020), based on a pedagogical framework for interpreting digital transformation in education, which views both teaching and learning as intertwined paths in multimodal contexts (Schlemmer et al., 2020).

In this paper, we provide clarification on how to follow the Inven!RA architecture specification to design and develop third-party activity modules. A simple case of a Web-based documentation download activity, and a more challenging case of a remote Arduino programming activity are presented. These modules underwent unit and integration testing with a prototype implementation of the Inven!RA architecture, provided by the research centre that developed it, INESC TEC in Portugal. The tests demonstrated the adequacy of the approach herein to design Activity Provider modules for platforms following the Inven!RA architecture specification.

2. Background

a. The Inven!RA architecture

The Inven!RA architecture aims to enable online tracking and orchestration of distributed learning activities provided by multiple third parties (Cruzeiro, 2020). It follows a broker pattern (Stal, 1995) inspired in the approach of the BEACONING project framework (Bourazeri et al., 2017), and was developed as a cooperation of the INESC TEC Associated Laboratory in Portugal and UNISINOS Project CAPES/PRINT "Digital Transformation and Humanities" in Brazil. It allows the design and tracking of distributed learning activities plans, called Inventive Activity Plans (IAPs), based on external third-party modules that provide each activity and record learning analytics about it. The Inven!RA architecture maps learning analytics of activities in an IAP to its learning objectives, and enables integration with learning management systems such as Moodle by providing customized URLs for teachers, for each learning activity.

The Inventive Activities Plans in the Inven!RA architecture consist of a graph of parameterizable learning activities, associated with specific learning objectives through a weighted combination of analytics (Figure 1). It is a distributed and agnostic architecture: the activities that form an IAP are conceptually external to the Inven!RA back-end and front-end: they are third-party components, accessed via Web services of entities labelled "Activity

Providers". This is one of the distinctive features of Inven!RA. Accordingly, the activities presented here as examples were envisaged as provided on the Web by third-party Activity Providers.

Education professionals take on two roles throughout the process. When designing a learning activity, the role of Learning Designer: crafting an IAP, specifying its learning objectives, assembling learning activities in a graph, providing configuration data for them, and mapping the activities' learning analytics with the IAP's learning objectives. When deploying an IAP, the role of Teacher/Trainer: using the IAP "as is" or adapting its configuration data and learning analytics mapping and considering the actual learners who will pursue it. Deployment is completed by Inven!RA, which requests each Activity Provider to create a specific instance of their activities included in the IAP, and generates a link for the Teacher/Trainer to include and configure in its preferred Learning Management System, as IAP entry points for learners.

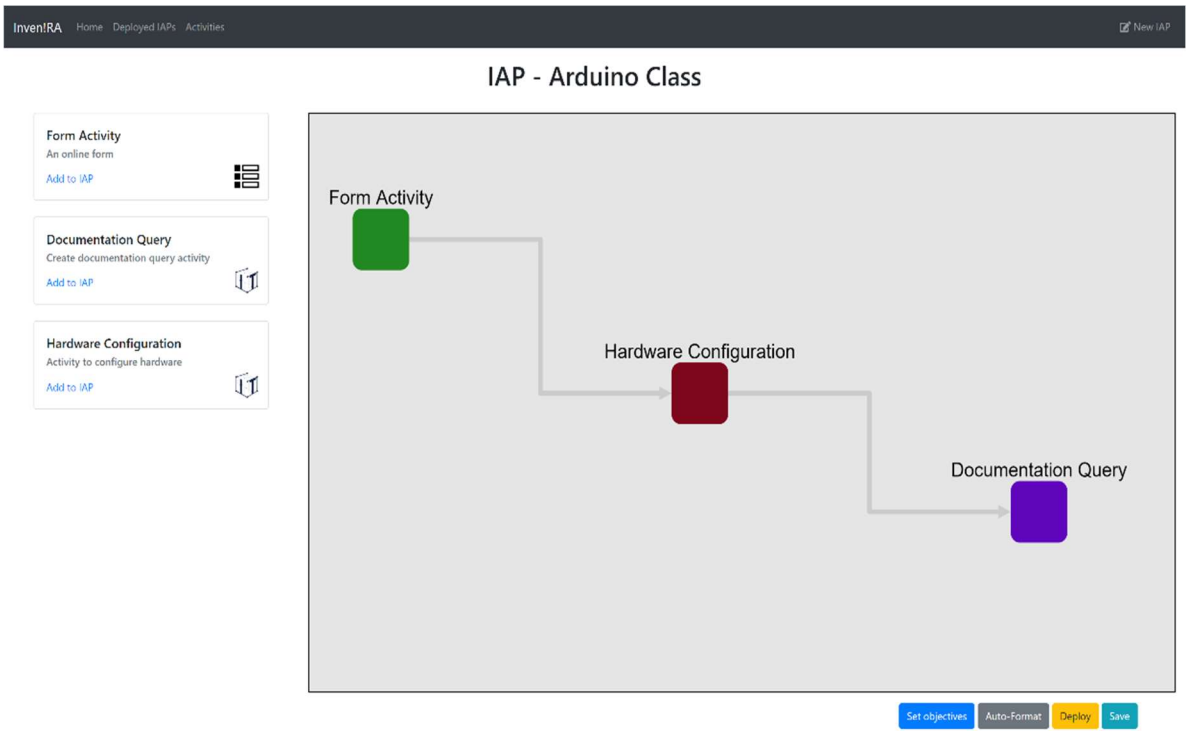


Figure 1 - Inventive Activities Plan on the default Inven!RA front-end (Cruzeiro, 2020).

As learners develop the IAP activities, learning analytics are collected by the Activity Providers, and the Teacher/Trainer can use Inven!RA to follow learners' progress towards the learning objectives, using analytics retrieved from the Activity Providers. Thus, the Teacher/Trainer can orchestrate its pedagogic intervention in accordance with the learning progress via the actual activities, instead of indirectly via tests or quizzes.

b. Inven!RA Activity Provider modules specification

The relationship between Inven!RA and an Activity Provider's activity starts when the latter registers the activity in Inven!RA, identifying it and indicating how that activity can be configured. This information is provided to Inven!RA as a JSON file as shown in Figure 2. The first couple of items are only for eventual front-end visual experience and not relevant to the backend operation (Cruzeiro, 2020, p. 47).

```
{
  "name": "activity name",
  "properties": [{
    "aprop": "smt"
  }],
  "config_url": "someconfigurl.html",
  "json_params_url": "web service returning params for harvesting from configuration page",
  "user_url": "url of web service to deploy activity",
  "analytics_url": "url of web service to request analytics",
  "analytics_list_url": "url of web service listing all available analytics"
}
```

Figure 2 – Inven!RA activity configuration JSON format

The relevant items are "config_url", which indicates the web service that should be called at the Activity Provider to retrieve the configuration Web page content; "json_params_url", which is a web service that returns JSON data listing all the parameters that Inven!RA can harvest from that configuration Web page; user_url, which is the web service that will be called to deploy this activity; "analytics_url", which is the web service that will be called to request analytics for this activity once deployed; and "analytics_list_url", which is a web service that returns JSON data listing all the analytics that the Activity Provider collects from the deployed activity.

The relevance of the "json_params_url" item is that it enables Inven!RA to harvest the configuration values of those parameters from the configuration Web page. These can thus later be provided to the Activity Provider when the activity is deployed. The "analytics_list_url" item is used by Inven!RA to enable the learning designer to map the activity's analytics to the IAP learning objectives.

When a Learning Designer includes the activity in an IAP, Inven!RA calls the "config_url" service, and embeds the results in its own activity configuration Web page. When the user completes the configuration, those values are harvested from that configuration Web page. Inven!RA used the list provided by "json_params_url" to identify them in that web page and stores them in the IAP for that activity. This process is shown in Figure 3.

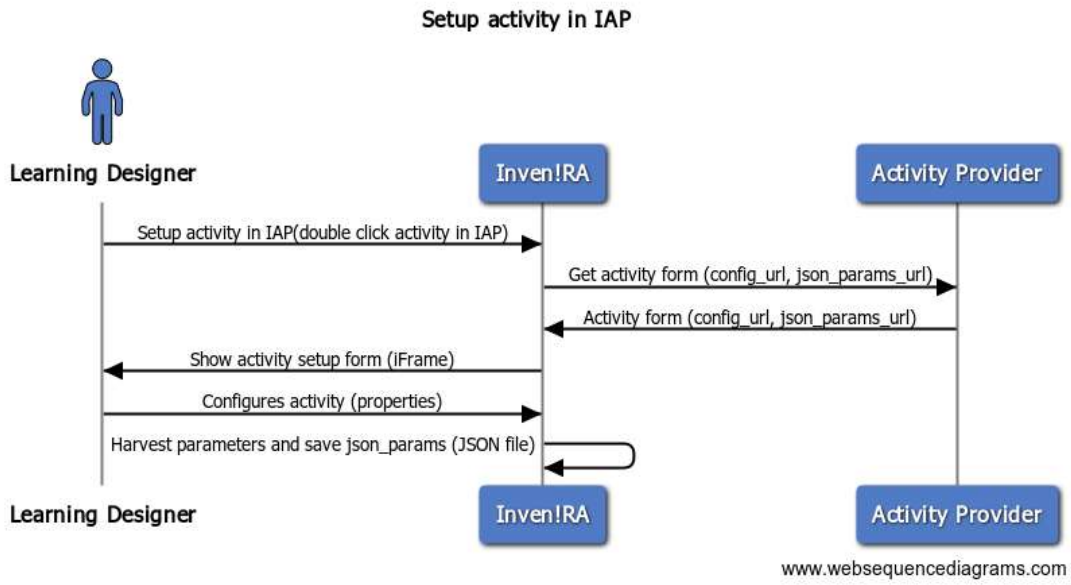


Figure 3 – Third-party activity inclusion in an Inven!RA IAP by the learning designer

After the Learning Designer sets up an IAP, configuring all activities and linking their analytics to the IAP learning objectives, the IAP can be deployed by teachers/trainers. When a teacher/trainer deploys an IAP for a class, an instance of the IAP is created. The config_url (Figure 2) is called with the payload of the json_params filled-in by Inven!RA with the stored configuration for the IAP. The teacher can accept this default configuration or edit it, with a similar workflow to that shown in Figure 3, but the custom configuration will be stored only in the deployed instance of the IAP.

The Inven!RA platform then calls the user_url service (Figure 2) with the activity ID that identified this activity as part of the deployed instance of the IAP. The Activity Provider responds with an URL to be used for deploying the activity. The Inven!RA platform generates an URL of its own associated with that one, to be able to act as middleperson between the students and the Activity Provider.

Inven!RA then returns to the teacher/trainer its URLs for all activities in the IAP instance, which the teacher/trainer can then include in its LMS, configuring it to provide studentIDs when those URLs are clicked. Henceforth, the flow shown in Figure 4 ensues: learners click on the URLs provided by the teacher, and Inven!RA receives those URLs requests. Inven!RA then replaces the LMS studentIDs with Inven!RA studentIDs to maintain student privacy, and forwards the final request to the Activity Provider deployment URL for that activity, along with the IAP id, and the specific configuration of that activity in the IAP (Figure 4, message "Get activity (activityID,Inven!RAStdID,json_params"). The Activity Provider is responsible for generating its internal instance and management of the activity for that activityID/user combination, and responding to the POST with Web page content, which Inven!RA will forward back to the student. From then on, the student will interact with the Activity Provider, identified solely by its Inven!RA student ID and IAP activityID, and the Activity Provider will keep the activity analytics. Any other interfaces (hardware, games, etc.) are dealt with from this pivot point.

Finally, when the teacher/trainer checks learning objectives' progress in Inven!RA, the platform requests the relevant analytics from the Activity Providers of the activities in the deployed IAP, by calling analytics_url (Figure 2), with the activity ID as payload.

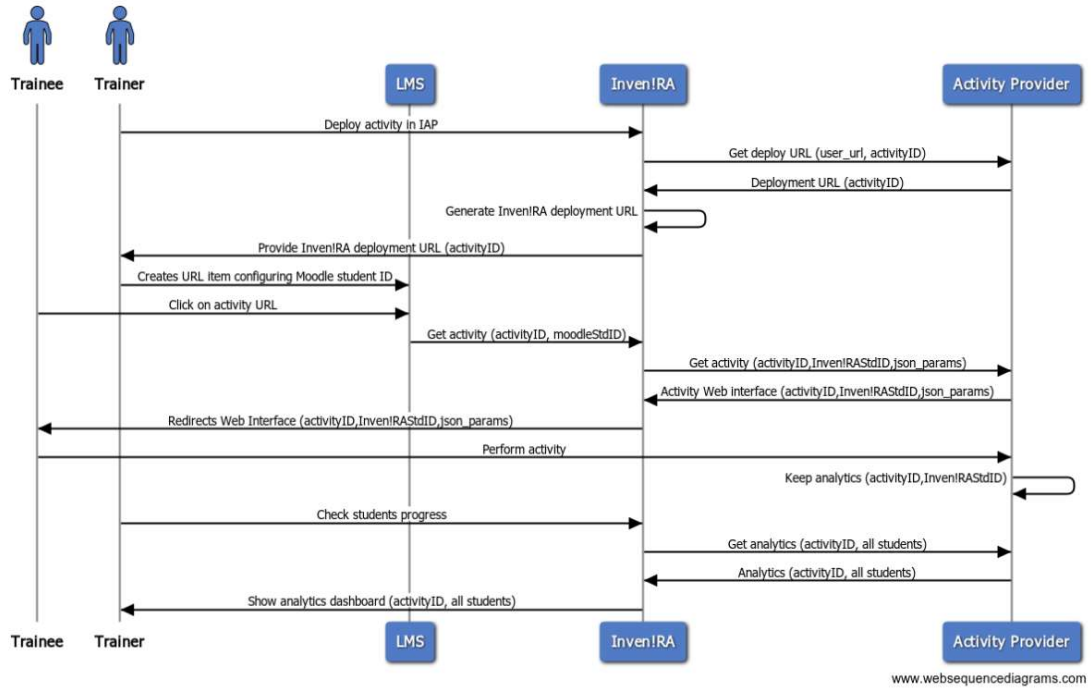


Figure 4 - Sequence diagram - Third-party activity deployment by Inven!RA

3. Selection of relevant cases

For clarification of design and development of Inven!RA Activity Provider modules, we sought to identify relevant learning activity cases that covered two scenarios:

1. A simple activity, to exemplify the core design.
2. An activity involving multimodal elements, to demonstrate avenues to pursue more diverse and complex activities.

As inspiration to devise these cases, we considered the Portuguese National Catalogue of Qualifications[1]. This catalogue specifies the learning contents and learning goals for vocational education and training courses in Portugal, called Short-Term Training Units (UFCD, Portuguese-language acronym). We chose UFCD 6073, "Microcontrollers - applications", which includes, among its objectives[2]:

- *Implement data acquisition and digital control systems.*
- *Program microprocessors/microcontrollers.*
- *Identify the key features of the microcontroller simulation and programming software being studied.*

[1] https://catalogo.anqep.gov.pt/ufcdPesquisa
[2] https://catalogo.anqep.gov.pt/ufcdDetalhe/5393

By combining the two desired scenarios with the list of objectives, we defined two relevant cases as targets to clarify design and development approaches for Inven!RA Activity Providers modules:

- Reading a technical description ("identify the key features").
- Programming an Arduino for data acquisition.

4. Scenario 1: Reading a technical description

This scenario, as expressed in section 3, represents a simple activity, to exemplify the core design. Students are required to read the documents comprising a technical description of the activity. We elected to implement it entirely as a single-page Web interface.

a. Architecture

Since the scenario follows the basic Inven!RA workflow shown in Figure 4, its architectural implementation is straightforward: a single message channel from the Inven!RA back-end to the Activity Provider back-end provides the three necessary services: configuration, deploy, and analytics.

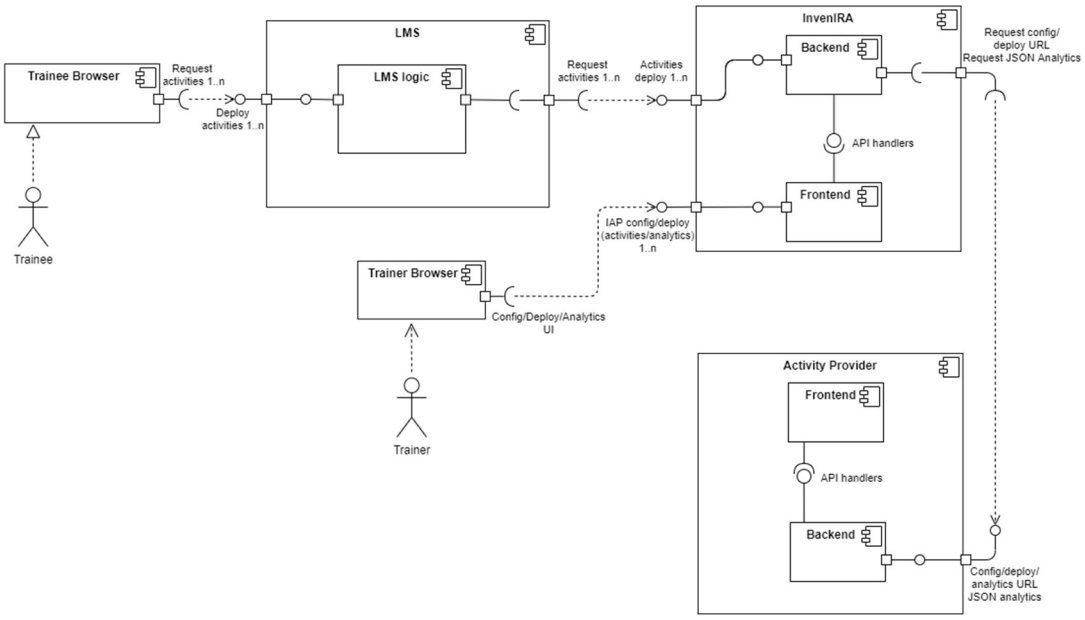


Figure 5 – Architecture component diagram for scenario 1

b. Implementation and use

The configuration Web page provided by this scenario is a simple form (Figure 6), enabling the learning designer to specify an URL to the main technical description document and a list of supporting reading as URLs with individual descriptions. It also contains a text field for inserting overall activity instructions.

Figure 6 – Configuration service output for Scenario 1

The sequence diagram for the document query activity deploy is shown in Figure 7, and is virtually identical to the basic Inven!RA workflow shown in Figure 4. For this activity, the "Perform activity" message consists in downloading the documentation to read, but otherwise all is identical.

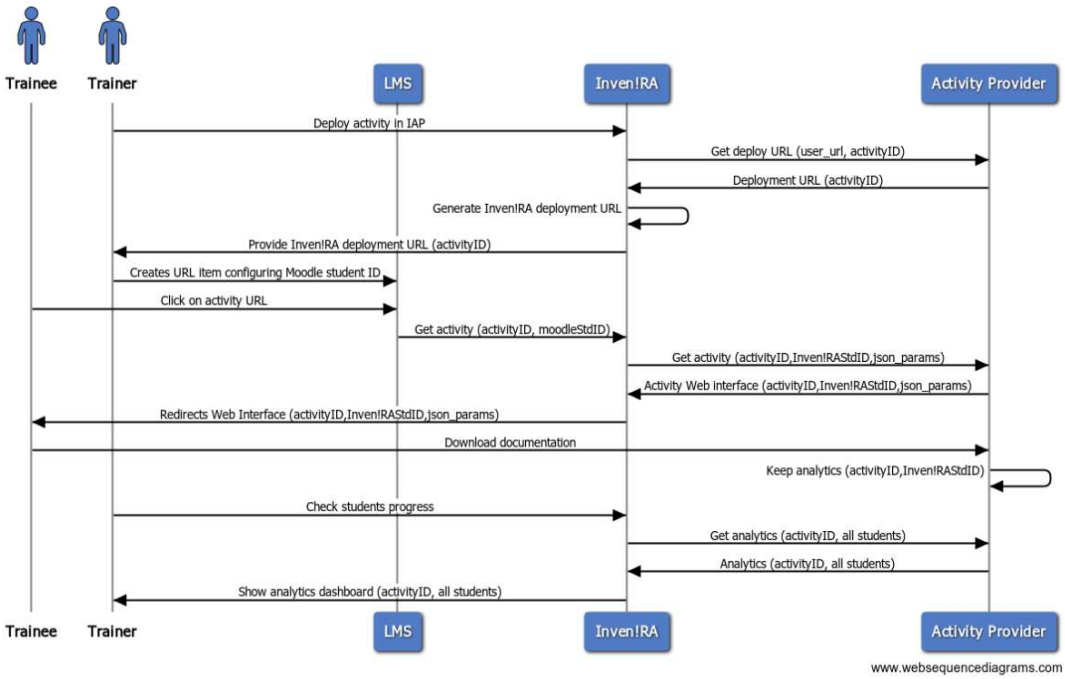


Figure 7 - Sequence diagram - Document query activity deploy

From the moment that Activity Provider receives the "Get activity" message, it must do whatever housekeeping and instantiation is necessary to provide that student with the activity and start collecting analytics for that student/IAP pair. The student interacts with the resulting web page (Figure 8) and any actions performed in it are recorded as analytics: access to the page and downloading of individual items.

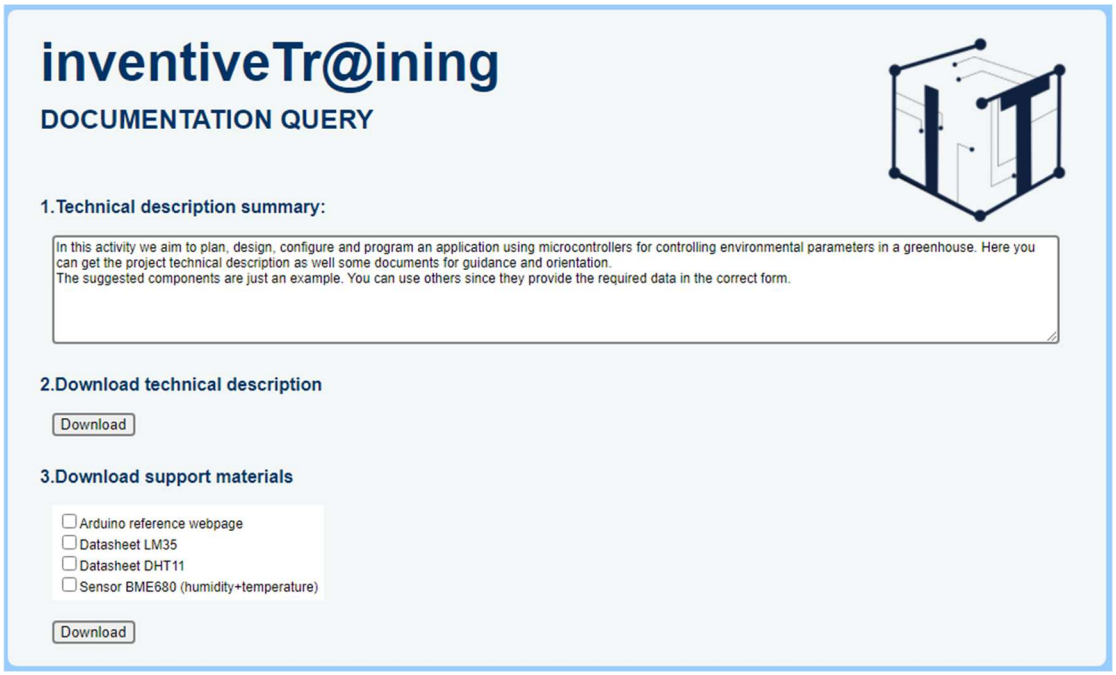


Figure 8 - Document query activity deployment

5. Scenario 2: Programming an Arduino for data acquisition

This scenario, as expressed in section 3, represents an activity involving multimodal elements, to demonstrate avenues to pursue more diverse and complex activities. The educational goal of this activity is for the student to test its Arduino firmware code, to evaluate the commissioning and operating conditions of its prototype Arduino hardware board. Hence, the deploy web page of this activity provides the trainee with instructions but also with a code skeleton, preset for Arduino board connection with the Activity Provider back-end via Ethernet or Wi-Fi, depending on the equipment previously delivered to the trainee. It is up to the trainee to encode the collection of data from board sensors, analyze it, and update its system outputs, according to the provided instructions.

a. Architecture

The role and interaction of the components in this scenario within the Inven!RA are shown in the components diagram of Figure 9. The Activity Provider module is seen on the bottom right, the Arduino hardware on the bottom left.

The main difference of this scenario is that the student must interact with an extra interface: an Arduino microprocessor board. This takes place entirely between the student and the activity provider, with no changes to the foreseen Inven!RA sequence shown in Figure 4. The analytics collection includes those related to the code uploaded to Arduino by the students and its operation.

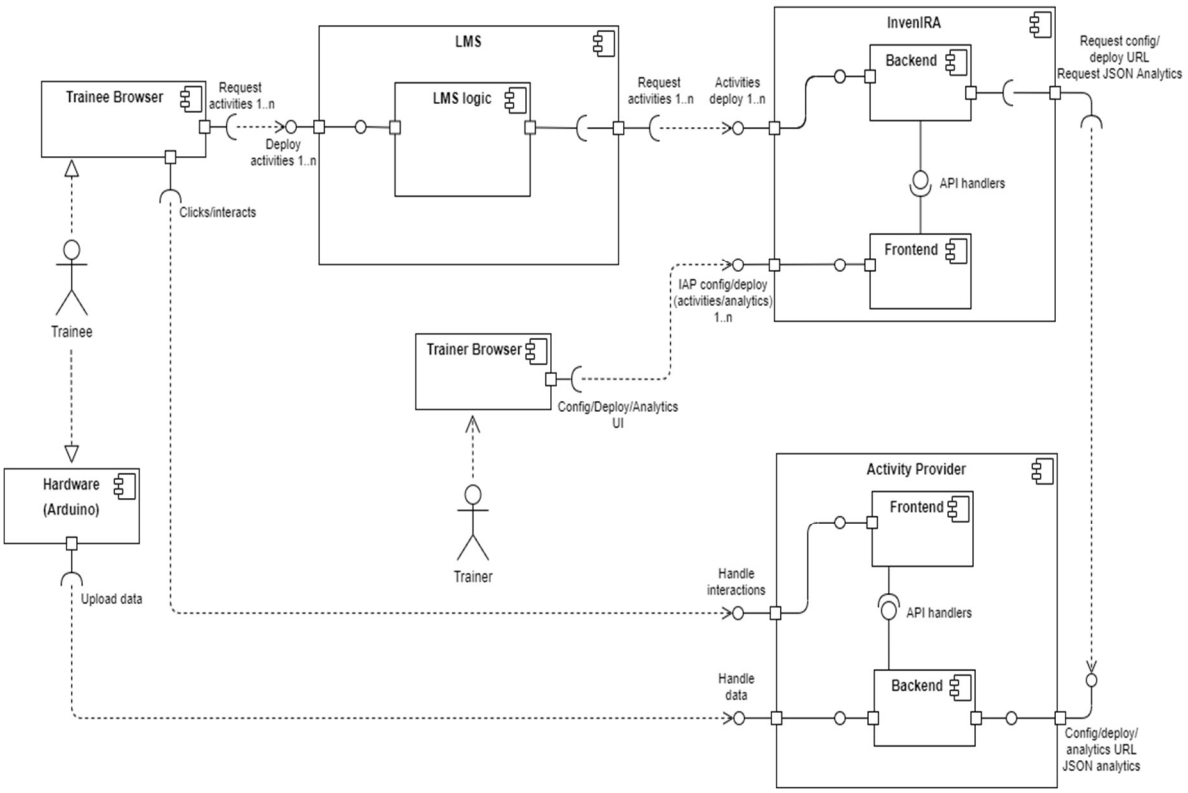


Figure 9 – Architecture component diagram for scenario 2

b. Implementation and use

In the configuration panel, the trainer provides activity instructions to the trainee, and technical documentation (e.g., sensor programming libraries). This results in the Inven!RA deployment presenting the student with all this information on a Web page, as with the Scenario 1 activity (Figure 10).

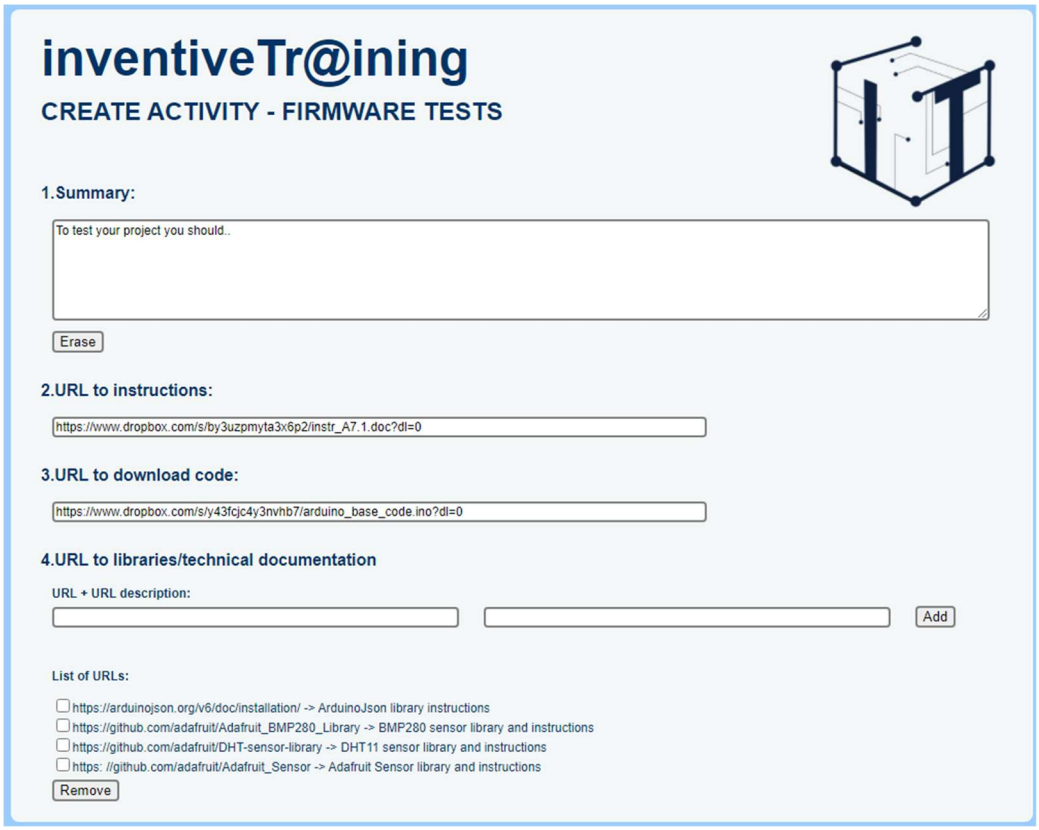


Figure 10 - Scenario 2 deployment Web page shown to students

A crucial element is item 3 in Figure 10: it is a link to a code template for the student to use on its Arduino board. This code template is automatically edited by the Activity Provider to include the individual studentID and IAP ID. When the student uploads it to the Arduino board and runs it, the code connects to the Activity Provider back-end, which can save successful upload of the template as an analytic. Henceforth, as the student updates its Arduino code, to make its board read data from sensors, the running code uploads new data (new progress analytics) to the Activity Provider.

This is the critical difference between this scenario and scenario 1, and is shown in Figure 11: the default workflow has an extra message from the student to the Activity Provider, “Process A”, detailed in Figure 12. That extra process consists in the students downloading and installing the firmware skeleton on its Arduino, and subsequently updating that code, while the code provides new analytics to the Activity Provider, associated with the student ID and the IAP ID.

This scenario could similarly be exploited to connect to Inven!RA activities with other multimodal interactions between the student and the Activity Provider: the channel “Process A” could be seen as providing external elements, such as a videogame, or another piece of hardware, with the means to identify the student/IAP pair associated with an activity, and to collect analytics henceforth.

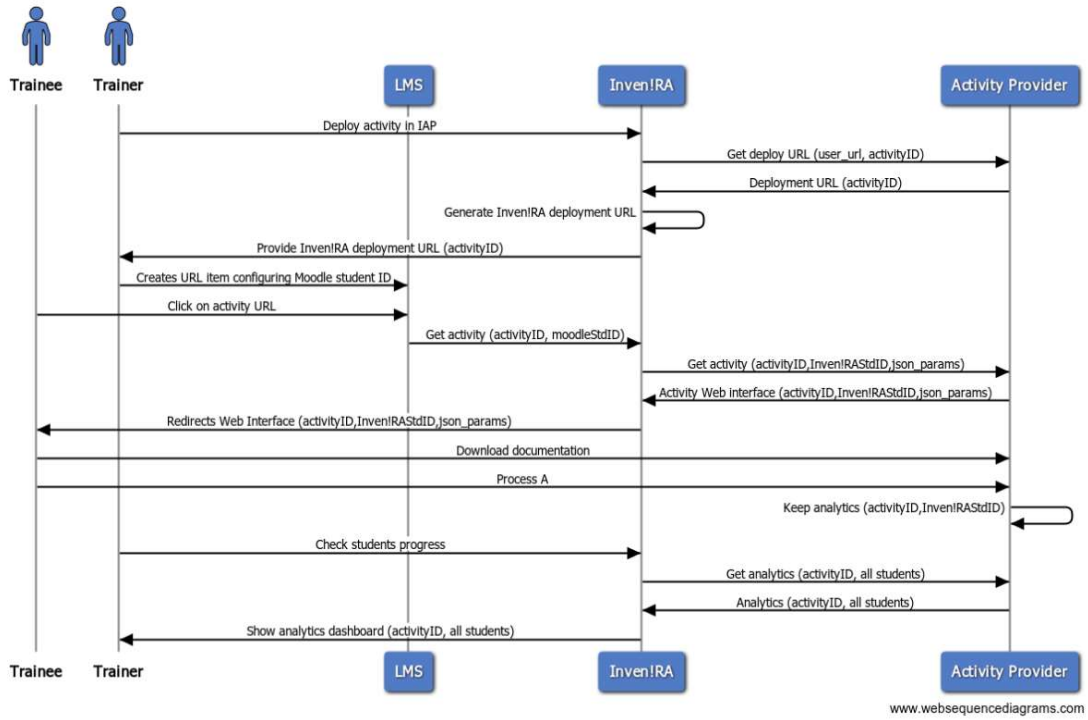


Figure 11 - Firmware tests activity sequence diagram

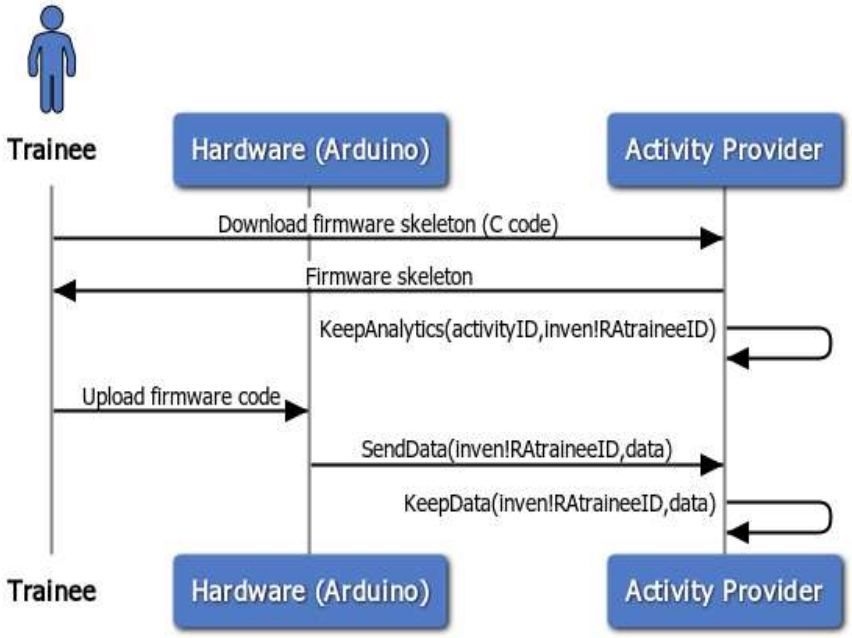


Figure 12 - Process A sequence diagram

In this scenario, the student is creating code to read sensors with the Arduino board. The skeleton code provides a standard message format for that, which is shown in Figure 13. Notice it includes the IAP ("activityID") and the student ID ("inveniraStdID").

```
From database {
  downloadDocsLibs: [
   {
     desc: 'ArduinoJson installation instructions',
     dLoad: true
   }
],
  studentData: [
   {
     temperature: [Array],
     pressure: [Array],
     alarm: [Array],
     setpoint: [Array],
     pressure: [Array],
     inveniraStdID: 1000
   },
   {
     temperature: [Array],
     pressure: [Array],
     alarm: [Array],
     setpoint: [Array],
     pressure: [Array],
     inveniraStdID: 1000
   }
  ],
  _id: 5edff325672e995388b4e1de,
  activityID: 1235,
  inveniraStdID: 1000,
  access: true,
  downloadInstr: true,
  downloadCode: false,
  upload: true,
  date: 2020-06-09T20:37:57.763Z,
  __v: 0
}
```


Figure 13 - JSON message sent from an Arduino to the Activity Provider server

In the example demonstrated in Figure 13, three test sensors were used to send data on air temperature, atmospheric pressure, relative humidity, and the status of an alarm that depends on a setpoint defined by the student in the coding phase.

The frequency with which such messages are sent is dependent on the actual project. In this example, we configured it to perform 10 readings with an interval of 2 seconds, to generate sufficient data for the trainer to monitor whether the objectives are being met by the trainee and, thus, whether corrections are necessary.

6. Implementation of analytics output

From the standpoint of Inven!RA, both scenarios are identical in terms of analytics operation: they collect data on student progress and actions, and associate those data with the student ID and the IAP. In the case of Scenario 1, these are simply related to access to the deploy page and downloading of its documents. In the case of Scenario 2, these analytics are also available, plus extra analytics about the progress of the Arduino programming, as shown on Table 1.

Table 1 – Analytics collected by at the activity provider for each scenario

Analytic	Scenario	Type
Student accessed to the activity deploy page	1 & 2	Boolean
Student downloaded technical descriptions	1	Boolean
Student downloaded the base code	2	Boolean
Student downloaded instructions	2	Boolean
Student downloaded the necessary libraries or technical support documents	2	Boolean
Student hardware ran the base code	2	Boolean
Student hardware sent test data	2	Boolean
Percentage of documents downloaded	1 & 2	%
Percentage of activity progress	1 & 2	%
Which documents were downloaded	1 & 2	Qualitative
Data sent by the student hardware	2	Qualitative

When appropriate, the trainer accesses Inven!RA and requests the IAP status. The platform sends a request to the Activity Provider indicating the ID of the intended activity or activities, and the Activity Provider server responds with the data collected for all students in that activity. Figure 14 shows a sample response. It includes the actual values for the quantitative analytics ("quantAnalytics"), and for the qualitative analytics, URLs are provided for Inven!RA to use ("qualAnalyticsURL"). Typically, Inven!RA will use the quantitative analytics to map students' progress toward the IAP learning objectives. The qualitative analytics can be provided as individual student URLs for the teacher/trainer to click, or shown embedded on a Web page, or employed for some other display alternative.

7. Integration tests with the Inven!RA platform

a. Registering activities

Currently, Inven!RA does not provide an automated process for registering Activity Provider modules externally, so this process simply consisted in contrasting the JSON format of Figure 2 with the format specified in the Inven!RA technical description (Cruzeiro, 2020, pp. 46–47 Fig. 5.3). This detected lack of a parameter in the original specification for the platform to request analytics, so one was added to Figure 2 ("analytics_url") and will be included in subsequent platform versions. We also detected explicit indication that the lists of parameters and analytics were retrieved from Web services, rather than embedded in the configuration file, so we renamed these items and their descriptions for clarity, as shown in Figure 2. Both scenarios were manually registered with their JSON files within Inven!RA, calling its appropriate internal microservice, to enable testing of the rest of the functionalities.


```
[
  {
    "inveniraStdID": 1000,
    "quantAnalytics": [
      {
        "name": "Access to activity",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Technical description download",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Technical documentation download (%)",
        "type": "percentage",
        "value": "75.00"
      },
      {
        "name": "Activity progress (%)",
        "type": "percentage",
        "value": "83.33"
      }
    ],
    "qualAnalyticsURL": "https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1000"
  },
  {
    "inveniraStdID": 1001,
    "quantAnalytics": [
      {
        "name": "Access to activity",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Technical description download",
        "type": "boolean",
        "value": true
      },
      {
        "name": "Technical documentation download (%)",
        "type": "percentage",
        "value": "100.00"
      },
      {
        "name": "Activity progress (%)",
        "type": "percentage",
        "value": "100.00"
      }
    ],
    "qualAnalyticsURL": "https://inventivetraining.herokuapp.com/qualAnalyticsDoc/?activityID=1234&inveniraStdID=1001"
  }
]
```


Figure 14 - JSON Response to analytics request

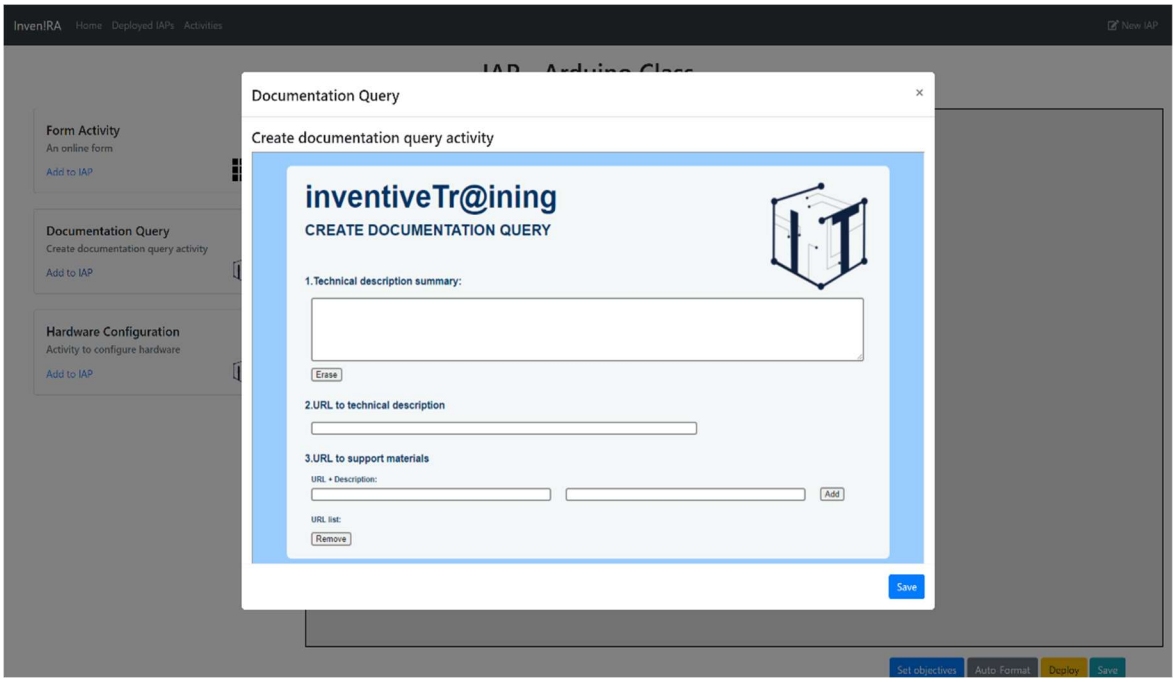


Figure 15 - Activity configuration integration test

b. Activity configuration

After registering the activities on Inven!RA, we proceeded with the integration tests for both scenarios. The first functionality (section 2.b) was the activities setup or configuration process. Using the Inven!RA front-end, the manually registered activities were dragged into an IAP. This resulted in the config_url parameter being called and responding with the matching HTML, which Inven!RA embedded in its configuration pane, as shown in Figure 15 (scenario 1, the same success occurred for scenario 2).

c. Activity deployment

After the learning designer completed setting up the activities in the IAP, a teacher/trainer could select it for deployment in Inven!RA. This process used the "name" property from our configuration JSON files as a visual content for the teacher. It also used the user_url parameter, which was called by Inven!RA for each activity, loaded with its activityID. As a result, the activity provider services generated URLs for the deployed activities. Inven!RA generated internally its own URLs to match those, and the activity names and Inven!RA-generated URLs were correctly shown to the teacher by the Inven!RA frontend, as shown in Figure 16.

The trainer proceeded with the standard Inven!RA procedure: placing these URLs in a Moodle LMS test course and configuring the LMS to attach user IDs when a student clicked them. By simulating user clicks in Moodle, then those URLs were called, and Inven!RA replaced the user LMS IDs with Inven!RA student IDs. Then Inven!RA called the matching Activity Provider deployment URLs with the relevant payload of data, as described in section 2.b. This process was simulated for the purpose of these integration tests. This call was received by the Activity Provider, which recorded internally the need to start collecting analytics for that Inven!RA student ID, and replied with the Web page that Inven!RA forwarded to the student, already shown in Figure 8.

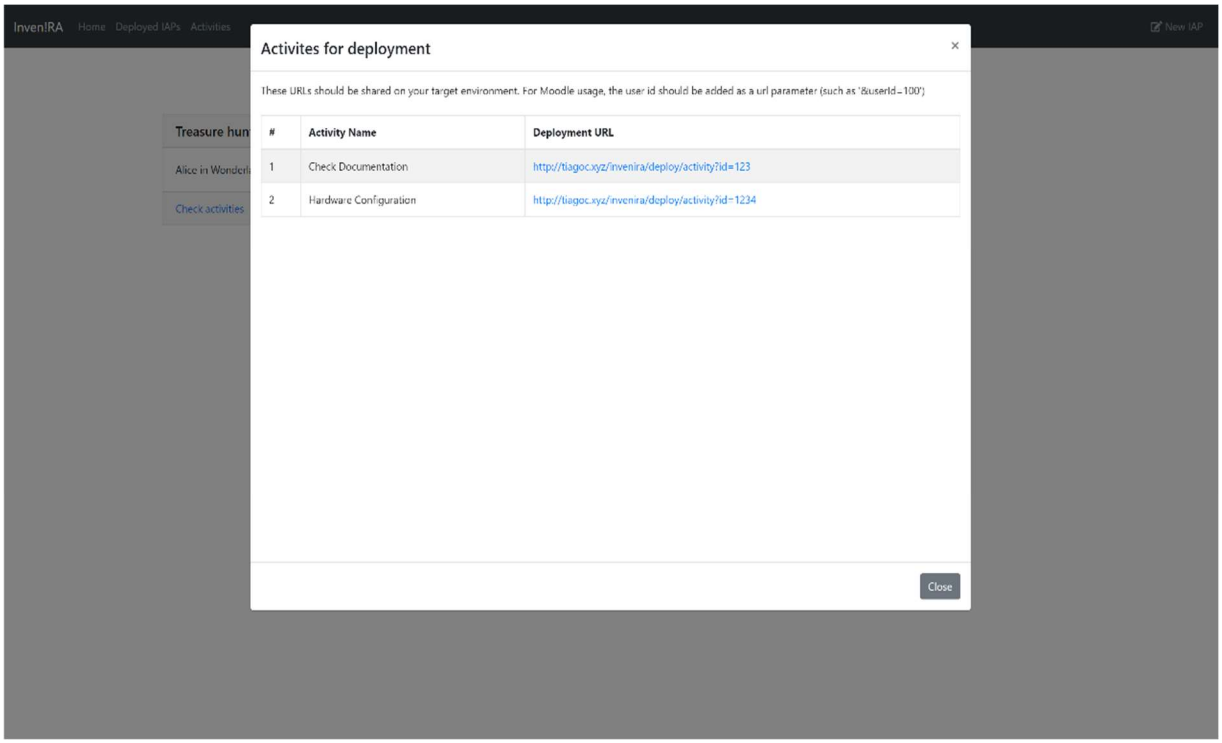


Figure 16 - Activity deployment screen in Inven!RA

d. Analytics

The deployment tests of the previous section were performed for Inven!RA student IDs 1000, 1001, 1002, and 1003. We requested analytics for both activities (scenario 1 and scenario 2) in the Inven!RA front-end. The requests triggered calls to the analytics_url of each activity. The Activity Provider correctly responded by sending the matching JSON files with the structure shown earlier (Figure 14). This file was processed by the Inven!RA front-end correctly, resulting in the output of Figure 17.

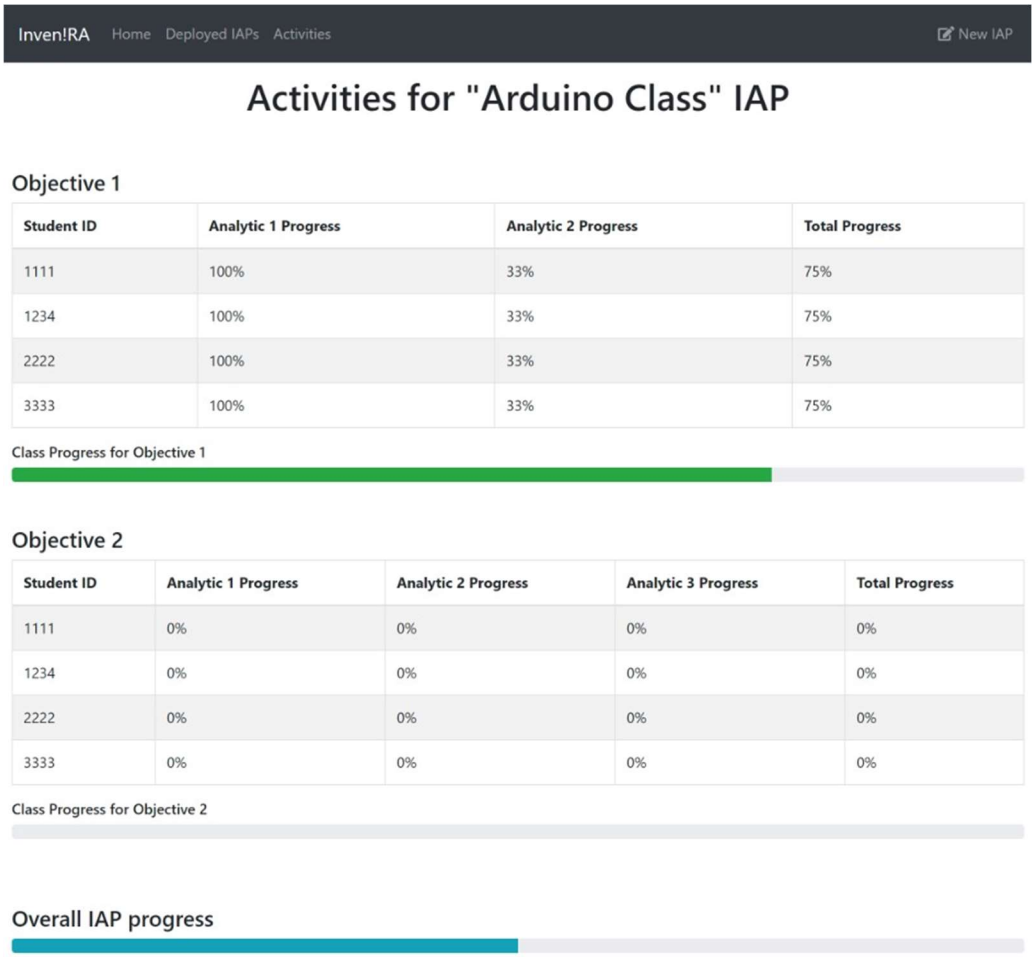


Figure 17 - Analytics query result for the activity of scenario 1

To access the qualitative analytics, the current Inven!RA front-end prototype does not yet provide a feature for their visualization. Hence, we created a mock-up front-end feature proposal (Figure 18). To test this proposal, we directly called the URLs that the Inven!RA backend received form the Activity Provider, which would be the URLs linked to the text "Check on provider" shown in the Figure 18 mock-up proposal.

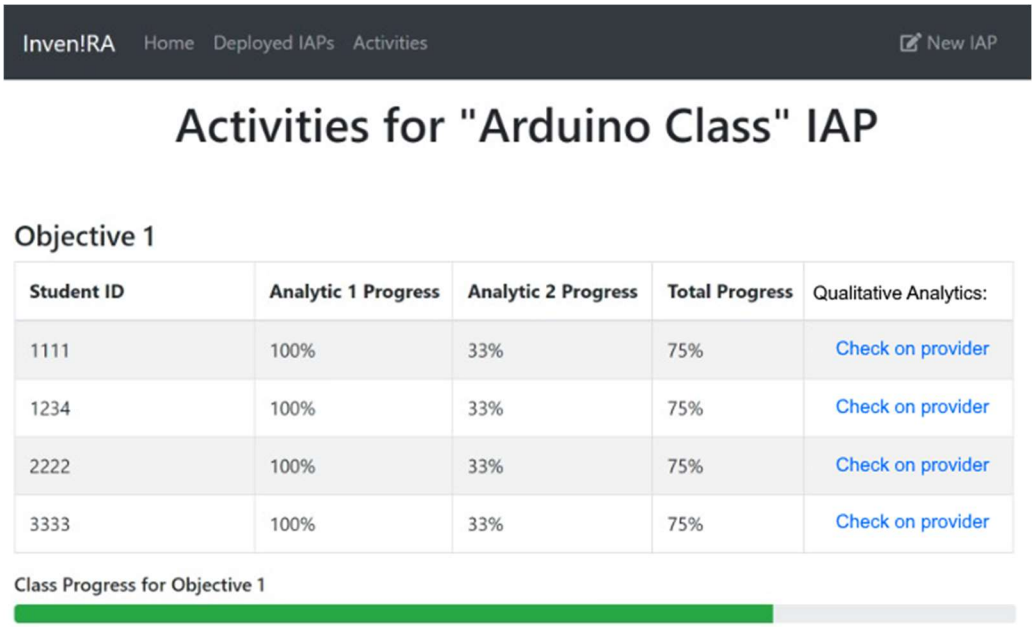
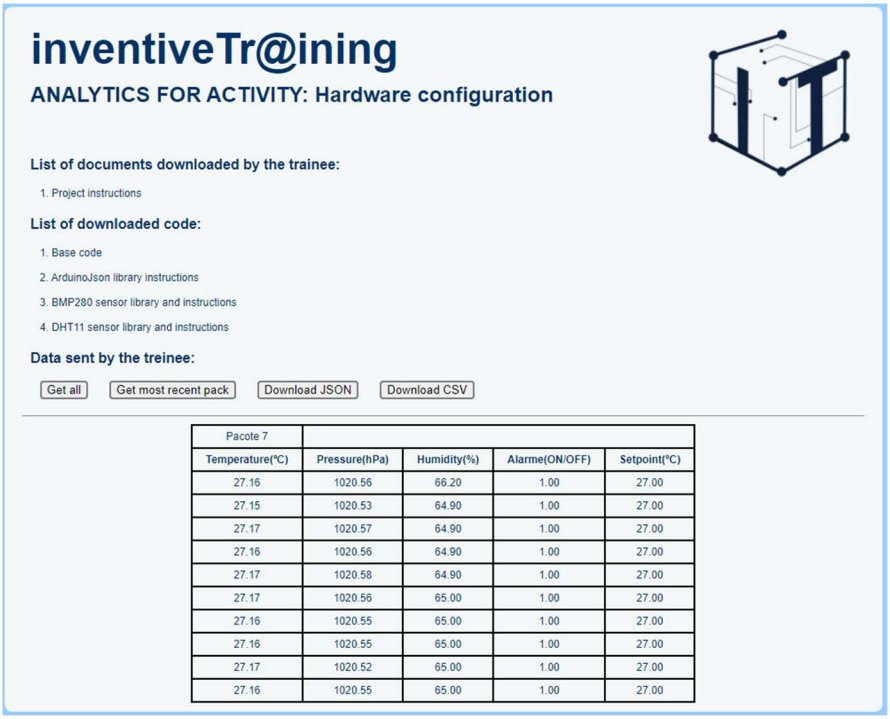


Figure 18 – Mock-up of proposed Inven!RA front-end solution for including qualitative analytics

The results of calling those URLs, in effect simulating clicking on the "Check on Provider" links for each individual student, resulted in the matching Activity Provider page for that student (Figures 19 and 20).

Figure 19 - Qualitative analytics page (scenario 1)

Figure 20 - Qualitative analytics page (scenario 2)



Pacote 7				
Temperature(ºC)	Pressure(hPa)	Humidity(%)	Alarme(ON/OFF)	Setpoint(ºC)
27.16	1020.56	66.20	1.00	27.00
27.15	1020.53	64.90	1.00	27.00
27.17	1020.57	64.90	1.00	27.00
27.16	1020.56	64.90	1.00	27.00
27.17	1020.58	64.90	1.00	27.00
27.17	1020.56	65.00	1.00	27.00
27.16	1020.55	65.00	1.00	27.00
27.16	1020.55	65.00	1.00	27.00
27.17	1020.52	65.00	1.00	27.00
27.16	1020.55	65.00	1.00	27.00

Figure 21 - Qualitative analytics page - viewing most recent data package sent by the student Arduino code

Pacote 0				
Temperature(ºC)	Pressure(hPa)	Humidity(%)	Alarm(ON/OFF)	Setpoint(ºC)
26.31	1021.62	71.10	0.00	27.00
26.31	1021.64	71.10	0.00	27.00
26.31	1021.65	71.10	0.00	27.00
26.32	1021.67	71.10	0.00	27.00
26.32	1021.62	71.10	0.00	27.00
26.31	1021.65	71.20	0.00	27.00
26.31	1021.63	71.20	0.00	27.00
26.31	1021.62	71.10	0.00	27.00
26.32	1021.62	71.10	0.00	27.00
26.32	1021.63	71.10	0.00	27.00
Pacote 1				
Temperature(ºC)	Pressure(hPa)	Humidity(%)	Alarm(ON/OFF)	Setpoint(ºC)
26.56	1021.67	78.40	0.00	27.00
26.55	1021.65	75.10	0.00	27.00

Figure 22 - Qualitative analytics page - viewing all data sent by the student Arduino code

In the test case of Figure 20, the simulated student would have already downloaded all the technical documents as well as the base code and instructions, as detailed on this page. The trainer can observe data sent by the trainee's Arduino code, using the various option buttons. Figure 21 demonstrates the result of the "Get most recent package" option, and Figure 22 the result of the option “View all data”.

This integration test has demonstrated that the analytics were indeed recorded at the Activity Provider for each individual Inven!RA student ID as identified during the deployment request, and adequately interpreted by Inven!RA for front-end display during the analytics request.

8. Conclusions

We have provided clarification on how to follow the Inven!RA architecture specification to design and develop third-party activity modules, using two relevant scenarios. Integration tests detected an omission in the original Inven!RA documentation, leading to its correction, and were successfully deployed. Thus, the adequacy of this approach was demonstrated to design Activity Provider modules for platforms following the Inven!RA architecture specification.

Future work should consider testing more diverse workflows involving teachers, students, and learning designers. For instance, to monetize the Activity Provider service, a contractual agreement needs to be established between paying users and the Activity Provider. This could consist of a token inserted in the activity configuration, either by the learning designer or the teacher, hypothetically resulting from a previous contractual phase. However, the feasibility

and complexities of such an arrangement have not been described by either the Inven!RA specification or the current paper.

Another open challenge is how to provide qualitative analytics at the class level. The current approach provides teachers with quantitative overview in the Inven!RA frontend, and simply stores links to individual students' qualitative information, which if available in the front-end interface would simply forward the teacher to individual students' details pages. Future work should consider exploring interaction between the Inven!RA front-end/back-end and the Activity Providers to enable class-level qualitative analytics support for the teacher.

Finally, student-side analytics have not been explored in Inven!RA at all. These would enable envisioning a world of widespread active learning using Inven!RA or similar technologies, not only by teachers but also by students, who require awareness of their ongoing activities and status, as individuals or teams, for adequate pedagogical orchestration, self-regulation and co-regulation of learning. This remains in our sights as an exciting research avenue.

While tackling these challenges, we support the need to maintain the distinctive goal of the Inven!RA back-end as an integration platform, which brokers activities between activity providers and LMS platforms, and does not intend to compete with existing LMS platforms. Also, we see as a critical aspect the need to maintain its separation of concerns between front-end and back-end modules, to complement diverse pedagogical orchestration needs with this brokerage ability, supporting the deployment of inventive learning activities that are attractive and stimulating for both teachers and students, facilitating their management and orchestration by teachers, the self-regulation processes by students, and the co-regulation processes by peers.

REFERENCES

Bourazeri, A., Arnab, S., Heidmann, O., Coelho, A., & Morini, L. (2017). *Taxonomy Of A Gamified Lesson Path For Stem Education: The Beaconing Approach*. https://doi.org/10.5281/ZENODO.1009096

Coelho, A., Rodrigues, R., Nóbrega, R., Jacob, J., Morgado, L., Cardoso, P., et al. (2020). Serious Pervasive Games. *Frontiers in Computer Science*, *2*

Cruzeiro, T. J. L. (2020). *Inven! RA-Platform for authoring and tracking of Inventive Activity Plans* [Universidade do Porto]. https://hdl.handle.net/10216/129252

Marklund, B. B., & Taylor, A.-S. A. (2016). *Educational Games in Practice: The challenges involved in conducting a game-based curriculum*. *14*(2), 122–135.

Schlemmer, E., Morgado, L. C., & Moreira, J. A. (2020). Educação e transformação digital: O habitar do ensinar e do aprender, epistemologias reticulares e ecossistemas de inovação. *Interfaces da Educação*, *11*(32), 764–790.

Stal, M. (1995). The broker architectural framework. *Workshop on Concurrent, Parallel and Distributed Patterns of Objects Oriented Programming, held at OOPSLA*, *95*.

Duarte Cota is a STEM educator at ENTA – School of New Technologies of the Azores and an IT teacher at the EST unit of the University of the Azores. He holds a BEng degree in Informatics Engineering by Universidade Aberta (UAb), the Portuguese Open University, and his final BEng project involved developing the Activity Provider modules presented here. He is attending a joint MEng programme in Informatics Engineering and Web Technology by the University of Trás-os-Montes e Alto Douro and UAb, with interests in those areas and Deep Learning.

Tiago Cruzeiro is a Senior Integrations Engineer at Farfetch, the leading global platform for the luxury fashion industry, connecting creators, curators and consumers. He hold an MEng Degree on Informatics Engineering and Computing by the Faculty of Engineering of the University of Porto. His MEng dissertation included the design and implementation of the Inven!RA architecture prototype.

Dennis Beck is an Associate Professor of Educational Technology at the University of Arkansas, and board member of the Immersive Learning Research Network. He enjoys teaching courses in issues and trends in educational technology, instructional design, integrating technology into the curriculum, and educational technology research. He also has a wealth of experience in the design of online and blended courses in educational and corporate training environments.

António Coelho is Associate Professor with Habilitation at the Department of Informatics Engineering of the Faculty of Engineering of the University of Porto, where he teaches in the areas of Computer Graphics, Programming and Digital Games, chairs the Doctoral Program in Digital Media, and is advisory board member of the UT Austin | Portugal Program. He is also a senior researcher at INESC TEC, coordinating the Computer Graphics and Virtual environments area.

Leonel Morgado is Associate Professor with Habilitation at Universidade Aberta (Portuguese Open University), a public university focusing on e-learning, where he lectures on programming, software patterns, and research methods. He develops his research at the INESC TEC research laboratory, on the use and development of immersive learning environments. Leonel is also vice-president for scientific quality of the Immersive Learning Research Network.

(esta página par está propositadamente em branco)