DOI 10.34658/9788366741102.7

Educational infrastructures for IT teaching

Marcin Kwapisz[0000−0001−5128−9009],
Michał Karbowańczyk[0000−0003−3875−1101]

Lodz University of Technology
Institute of Information Technology
Wólczańska 215, 90-924 Łódź, Poland
marcin.kwapisz@p.lodz.pl, michal.karbowanczyk@p.lodz.pl

Abstract. *The basic problem of teaching information technologies is a proper infrastructure for classes and courses. This especially applies to remote and e-learning, when a student does not have access to "physical" laboratories. Technologies based on virtualization and containerization can help a lot to deliver the infrastructure. Unfortunately it requires a significant amount of administrative work, because many tasks simply have to be done manually. This article explains the current situation for Infrastructures and Network Applications (INA) specialization at Institute of Information Technology. It also describes how the educational infrastructure could or should be managed in the future.*
Keywords: *education, computer systems, infrastructures*

1. Introduction

One and a half year of remote learning upon pandemic restrictions resulted in a significant change to the teaching at the Institute of Information Technology. The transition to remote mode for the INA specialization was carried smoothly and seamlessly. The method of access to the infrastructure required for our classes and courses (courses in short) has not changed at all. The whole infrastructure is built on the basis of the physical equipment located in the LAB108 of Information Technology Center (CTI) and VMware vSphere cluster installed on top of it. From students and teachers point of view only a workplace has changed from laboratory rooms to home desks. All courses have been created on the WIKAMP platform from the very beginning. The only noticeable problem was lack of direct contact with students during lectures when a teacher cannot observe them and their reactions. Less meaningful problems occurred during laboratory classes. In smaller groups students could turn on their webcams and teachers could have face to face conversations, sort of at least. In this article authors would like to display the state of the INA specialization, describe currently used tools and problems with them, and eventually make an attempt to find solutions for these problems.

2. Set of courses for the IAS specialization

2.1. Semester V

Specialization courses start at the 5th semester of engineering studies cycle. They constitute one coherent and linked together sequence of courses. Very often students are required to use previous effects of their work during the next semester. The sequence starts with two core courses:

- Infrastructures of Development and Production Environments (IDPE),
- Fundamentals of Network Applications (FNA)

and are supported by two additional courses which results can be used during the next semester as an add-on in a form of a remote client or a control element for a business system:

- Introduction to Mobile Systems,
- Fundamentals of IoT systems.

The aim of both core courses is to prepare students to organize their development and deployment environments also with high availability in mind. The second course is a typical introduction to architectures, design and development of web applications. Students have the opportunity to get acquainted with both monolithic and distributed business applications. Additionally, they are introduced to basic security mechanisms like authentication, authorization and network security.

2.2. Semester VI

During the VI semester students have three more core courses and one complementary:

- Basics of Virtual Systems (BVS),
- Technologies of Network Components (TNC),
- Network Database Systems (NDS),
- Fundamentals of security of internet systems (FSIS).

The BVS course is based on the standard VMware vSphere Install, Configure, Manage v7 course and partially on Optimize and Scale v7. Students have the opportunity to learn about functionality and mechanisms delivered by VMware vSphere and VMware VCenter Server which are main constituents of all cloud systems. While the first subject is typically infrastructural, the other two relate to software design and development. TNC course is strictly connected to microservice architectures, technologies and network communication. Students have to apply communication methods and protocols, mechanisms for external configuration delivery and monitoring of applications. NDS is focused on working in larger teams on a development of a fully functional DBMS-based web application. Students have to uses their knowledge and skills acquired in the previous semester, as well as during parallelly led courses. The complementary FSIS is as important as the core courses. While authentication and authorization provide the basic security for networked systems, students must be aware of existing internet threat, how to circumvent and counteract it by doing for example security scans with generally available tools.

2.3. Semester VII

There is only one core specialization course during that semester: Application Maintenance Techniques (AMT). It is designed to teach students about techniques of maintenance and deployment of distributed application. Students have to deliver configurations and deploy their applications in containerized environments like **OpenShift**. Applications have to be:

- monitored,
- resilient to infrastructure and software failures,
- scalable to handle additional load.

3. Current infrastructure

In the section 2 the set of courses was described. They require proper infrastructure to allow students to run their activities, do exercises or to develop and deploy their projects. It consist with three main functional elements and several additional components build on top of computer cluster with VMware vSphere and VMware vCenter Server installed:

- Atlassian platform: **Jira**, **Confluence**, **Bitbucket**, **Bamboo** [1],
- **Netlab** [2],
- **OpenShift** [3].

By default all courses mentioned here are placed on WIKAMP educational platform.

3.1. Atlassian

All software development courses are supported by Atlassian platform which consist of:

- **Jira** - agile projects issue and tracking system,
- **Confluence** - teams collaboration and documentation,
- **Bitbucket** - source code management,
- **Bamboo** - continuous integration and deployment (CI & CD [4]),
- **Crowd** - single sign-on (SSO) for entire platform, accounts and groups management.

All Atlassian applications are integrated with each other. Information stored therein is shared, and many operations in one of them can be initiated from the other. Table 1 displays which courses use the Atlassian platform. Object Oriented Programming (OOP) and Component Programming (CP) were added as examples of mass courses what had great impact on presented design of a management layer in the section 5.

Table 1. The Atlassian platform usage

Course name	**Sem.**	**Bitbucket**	**Jira**	**Confluence**	**Bamboo**
OOP	II	X			
CP	III	X			
IDPE	V				
FNA	V	X			
BVS	VI	X			
TNC	VI	X			
NDS	VI	X	X	X	X
AMT	VII	X			

First changes were introduced to Atlassian platform (Atlas) in 2020 to comply with more modern IT solutions. From technical point of view Atlas was and still is a virtual machine that is run on the VMware cluster. Each of the Atlas constituents has been run as a separate instance of application server before. It was completely rebuilt then. Atlassian provides **docker** images with applications already installed. Now Atlas applications are run as containers under **podman** as a containerization manager. Application update process is much easier to perform as images are generally available on the Dockerhub. Directories with applications data and the database itself remained mostly untouched. The database system also runs as a container of course. Thanks to **podman** and the ability to define networks, it was possible to isolate all applications in one network. Applications communicate on this network using their container names. However, students need to access them somehow. There is additional container with nginx working as a reverse proxy server. It is the only container with exposed and mapped http/https to the host (atlas virtual machine) ports. What is worth noting, this is exactly the same solution that students are taught during IDPE and AMT courses. Containers have made administration of this server much easier and less time consuming. The general architecture of Atlas is depicted on Figure 1

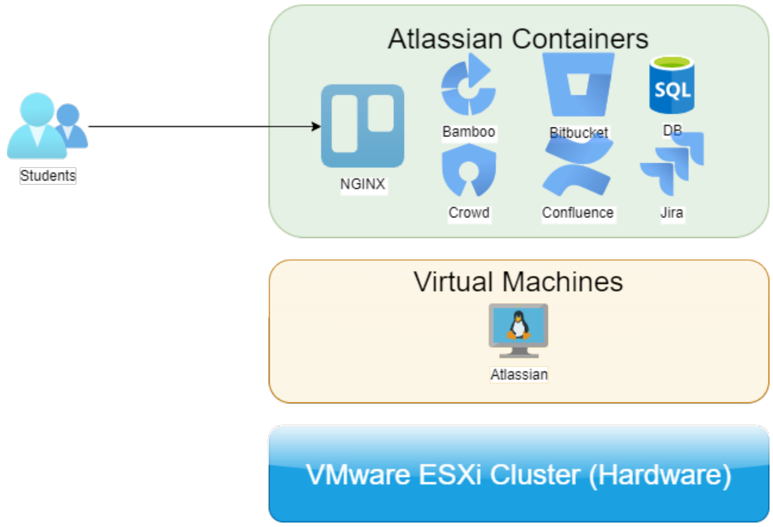

Figure 1. The Atlas architecture

3.2. NETLAB

This is the second platform that is used for educational purposes. As Atlas, it is based on **VMware vSphere** cluster and **VMware vCenter Server**. A teacher has to configure a set of virtual machines (pod) that will constitute a laboratory stand template. It can be self-prepared or ready-made by external company, like for example NDG (the developer of **NETLAB**). **NETLAB** itself is a virtual machine, but it can communicate with the vCenter Server which manages the entire VMware vSphere cluster. Thanks to granted permissions NETLAB is able to create, start and stop virtual machines, create virtual networks and add virtual machines to them. New pods are created from the template pod and they can be configured as stateless or stateful. However, the most important function of **NETLAB** is provisioning of tools that gives access to desktops of these virtual machines. This is a kind of a Virtual Desktop Infrastructure (VDI) The following courses use the NETLAB platform:

- BVS,
- FSIS.

As it was mentioned in the section 2.2, BVS uses redy-made pods prepared by NDG company. Figure 2 depicts NETLAB architecture.

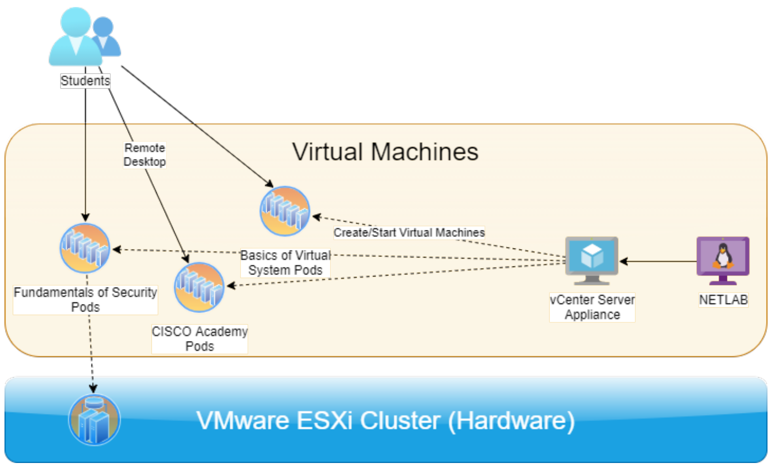
Students
Remote Desktop
Virtual Machines
Create/Start Virtual Machines
Basics of Virtual System Pods
Fundamentals of Security Pods
CISCO Academy Pods
vCenter Server Appliance
NETLAB
VMware ESXi Cluster (Hardware)

Figure 2. NETLAB architecture

3.3. OpenShift

This is the newest part of our educational infrastructure. It was preliminary installed for the AMT course. And what is explained later in section 5, it will be used also as an environment for a management platform. During the AMT course students have to migrate their previous projects (mainly from NDS course) to a containerized environment which **OpenShift** provides. It is not so easy, as it could be at the first glance. The deployment process must be done from sources from git repository or from an image stored in an image repository, like e.g. Dockerhub. Students have to deliver:

- source to image scripts,
- deployment descriptors,
- pods descriptors,

- routing information,
- security elements like certificates and secrets,
- an external container configuration like database or message broker connection parameters,
- and configure monitoring and autoscalers for their applications.

The required knowledge comes from previous courses like TNC and BVS. Openshift is going to replace actual infrastructure for the NDS course. Students teams require working Payara Application servers [5] (Java/Jakarta EE [6] compliant) and **PostgreSQL** database servers [7]. A pair of mentioned servers must be accessible to only one team. It can be delivered in the form of OpenShift project with two running containers. Now there are two virtual machines on the VMware vSphere cluster with a single installation of the **PostgreSQL** server and the **Payara Application Server**. In the case of the application server one of its internal mechanisms was used - division into the so-called domains. Despite using the same server instance for many teams, a configuration and management of deployed applications is limited to one domain. In the case of the database server the problem is solved by creating independent database instances within it.

4. Current administrator point of view

Let's define a term "administrator", because it may suggest we have "normal" administrator employed. This role unfortunately applies to teachers who have to manage all mentioned in section 3 systems and applications, including the VMware vSphere cluster of course. We simply cannot focus only on teaching but we have to do a lot of time consuming administrative activities, actually we do not want to do. Let's consider and analyze main problems with currently used systems and applications.

4.1. Atlassian

Atlassian Crowd fulfills the role of an authentication system, thus is used during the whole studies life-cycle and is used by the whole Atlassian platform. This is the most serious disadvantage of this solution. Lodz Univeristy of Technology (LUT) has its own authentication system based on **Jasig CAS**, which cannot be easily integrated with **Crowd** for several reasons: lack of control over application permissions (application authorization, see table 1), lack of control and configuration options over the user database. **Crowd** stores not only accounts, but also groups through which users gain access to a given Atlassian application or its resources. The second very serious problem is creation and removal of students' accounts. Accounts are created only once for given student during the second semester for the Object Oriented Programming course. Every year the "administrator" has to generate almost 200 accounts for the 1st cycle students and assign them to proper groups for courses they attend. This is the second most time consuming activity, even though it is performed by self-developed Java application. Data files for the Java application are exported from the WIKAMP "assign to a group" assignment. The textual file format is easy to process and contains account information like student id and a name of course group the student belongs to. If an account exists, the Java application adds an existing account to a selected course group only. A course group is than added to an application group, what gives access to selected Atlassian application like Bitbucket for example. To conclude that part, Crowd stores now over 1000 active accounts, 10 course groups and 12 application groups (e.g. for Jira there are 3 different application groups with different access levels: admins, developers, users).

The Java application also supports administrative operations on Atlassian Bitbucket. They are the most time consuming throughout the entire platform. All courses use git repositories, including mass one like OOP and CP. For 200 students a hundred of repositories is required, assuming that students work in pairs. An administrator has to remove previous academic year projects and their repositories, then create new projects and demanded amount of repositories and last but not least assign to them "write" rights for students. It would be impossible to perform that process manually in a reasonable time boundaries. The Java application made git repositories managed by Bitbucket server available for each course led on our specialization (and not only). Thanks to the automation process delivered by the Java application, creating hundreds of repositories takes hours but not days. The Java application uses Crowd and Bitbucket RestAPIs to perform all operations. It requires from the "administrator" a little of manual work to do, like export of required data from WIKAMP course and provide input parameters for the application. From "administrator" point of view, these operation should take no time at all to allow him to focus on teaching, not administration.

Jira, **Confluence** and **Bamoboo** are out of the scope of the Java application. The automation process was never implemented for them due to small volume of projects. They are created by teachers and the most affected course is NDS where all Atlassian application are used (see table 1). Even though the number of project is small, the whole process is very complex. **Jira** and **Confluence** projects are created from templates to deliver a kind of a starter pack to teams. Of course teachers have to assign proper rights to these projects (like for **Bitbucket** repositories) and create application links between Jira, Confluence, Bamboo projects and **Bitbucket** repository to form fully functional development environment.

4.2. NETLAB

This is a completely separate system to Atlassian platform where a teacher has to create a class, accounts and assign them to a class. Required account information can be exported from a WIKAMP course and imported to **NETLAB** using its web user interface. As for Atlassian administrative processes this is a very time consuming task. It is done once per semester but teachers would like to avoid that "creative" job. The biggest problem with **NETLAB** is pods creation. Luckily a teacher has to create only one template pod (set of virtual machines). This operation engages two systems: NETLAB and vCenter Server, and is performed by scripts written by PhD Eng. Michał Morawski. Working pods must be created as a linked copy of the template. Virtual disks of working pods are differential, so they do not take much place on physical storage. Despite that additional tasks have to be performed manually like: pods distribution to specific VMware vSphere cluster nodes. **NETLAB** does not support VMware vMotion and load balancing to do it automatically. It assumes that a pod is already placed on chosen node and just starts it up. As Atlassian applications, **NETLAB** provides an RestAPI through which most administrative tasks can be performed.

4.3. OpenShift

A 6-node **OpenShift** cluster consisting of 3 worker nodes and 3 control plane nodes has been installed on the VMware vSphere cluster. The number of cluster nodes is not accidental. One **OpenShift** node runs on one VMware vSphere cluster node. Three control plain nodes should provide at least good fault tolerance level for the entire cluster, but it occurred that it is not true. Its ETCD database which stores the cluster configuration was completely destroyed two times on all three control plane nodes by power failures. So the cluster had to be re-installed completely and restored from a backup.

Once again, as for the other systems, teachers have to create students accounts and projects. Accounts can exist as local accounts, or OpenID Connect [8, 9] server accounts (OIDC - e.g. **KeyCloak** implements it). What is interesting **KeyCloak** can be easily installed on **OpenShift** platform itself using an operator. Unfortunately, there is a serious flaw in the **OpenShift** platform because it cannot read group information from tickets received from an OIDC server and automatically import it to its local database. This drawback has effectively eliminated the use of OIDC. An attempt was done to eliminate two of three user databases (from **Atlassian Crowd** and **OpenShift**) and replace it with one. OIDC is a standard internet protocol and authentication mechanism for internet applications. Accounts and projects are not enough. An “administrator” has to put some cpu and memory restrictions on students’ projects. Otherwise they could easily exhaust all compute and storage resources or run malicious software like cryptocurrency miners.

4.4. Application servers and RDBMS

This is the next system that requires creation of students accounts. An access to two separate remote terminals to linux virtual machines must be granted to manage a domain on the application server and an instance of a database. It is possible to use the **Atlassian Crowd** server as an authentication mechanism and thus avoid defining another user database. However this approach implies further problems related to mutual separation of activities of project teams. While it is possible to solve them in the context of data separation (through appropriate use of authorization systems), the problem of sharing computational resources still needs to be solved. In the current state server resources can be easily exhausted by the activity of one of project teams. Please note that it is a virtual machine. All these problems should disappear after transition to **OpenShift** platform where there are three worker nodes (virtual machines). **OpenShift** can evenly and automatically distribute the whole workload among them.

5. Missing management layer

Lets conclude it a little bit:

- current educational infrastructure has outgrown teachers administrative abilities, even though they are supported by self-developed applications and scripts,
- there are three different user databases that cannot be integrated into one (**NETLAB** does not support OIDC) and two additional virtual linux servers for the NDS course,
- different applications or systems require different kinds of projects and their configuration.

The central administration model simply is not working. Fortunately all three systems deliver RestAPIs that can be used and the two linux systems can be transformed to **OpenShift** containers. Administrative scripts and the Java application use RestAPIs but in a very limited way. Still many tasks have to be done by teachers manually what consumes their time. The only solution for that problem is to transfer administrative tasks to students. It can be done but requires additional component that is missing now.

The model in which an administrator provides the necessary infrastructure or services has passed in a distant past. Cloud systems have introduced an infrastructure or service on demand a long time ago. Of course from IT perspective. Students could use generally available Microsoft Azure, Amazon AWS, Heroku or Google Cloud, but only the smallest deployments with restricted functionality are for free. The variety of different platforms students can use will make project assessment a real nightmare. The educational platform should be consistent across all courses and well known to teachers. All cloud systems are based on additional software layer that manages underlying physical or virtual infrastructure. Exactly the same must be done in our case. Therefore it was decided to initiate a project and develop the missing component or rather it should be named a management layer. It is in the inception phase now but several requirements were elicited:

- only students' accounts should be created, integration with LUT Jasig CAS authentication system is in a sphere of wishful thinking and a little problematic,
- a list of required resources (BOM, bill of materials) is configured once for a course,
- inversion of control - students create required resources for themselves using the management layer. Student chooses a course and the management layer communicate with **NETLAB**, **Atlas** or **Openshift** using their RestAPIs to perform operations described in the section 4 and deliver BOM,
- The list of available courses in a given semester is taken form the study cycle program. It is easy to acquire that information by scraping *programy.p.lodz.pl* web page.

Of course there are details to be considered like a student with an individual study program or a team project. The first one is rare and from administration point of view irrelevant. The second is more common, but project resources can be created once by a team leader. Then the leader can grant access to the rest of team members. Of course the rest of team members could request and create resources for themselves, but we think it will be a very isolated case.

The general architecture for the management layer to be developed is depicted on Figure 3

6. Conclusions

The INA specialization for 1st cycle studies at our faculty was briefly presented here, as well as the infrastructure built and maintained by teachers who lead

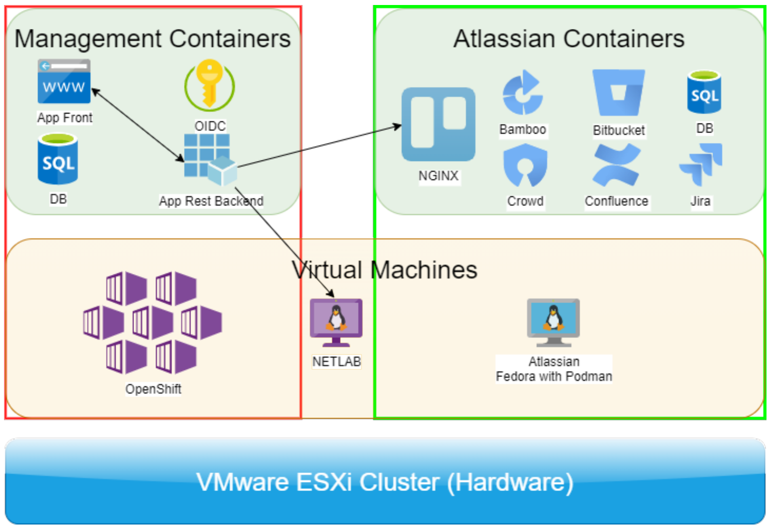

Figure 3. General architecture for the management layer

courses within it. The main problem is the lack of consistent management system for administration of the infrastructure . Tools facilitating administration created by teachers significantly reduced time needed for that. However, these tools allowed application of the infrastructure (**Atlassian**, **NETALAB** and **Openshift**) to much wider set of courses and mass courses for hundreds of students like OOP and CP as well. Easy to predict result was increased time for administrative tasks and overloaded teachers. This has to be stopped and an idea for a system appeared where students through the management layer will automatically create required resources for themselves. Only a set of resources should be defined by teachers as BOMs. The managing layer will store student accounts and, if necessary, create them in the managed subsystems. This should completely free teachers from "unnecessary" tasks and allow them to focus on development of their courses.

What is considerably important the management layer will be developed by teachers. The techniques and technologies presented during specialization courses will be applied and used. The infrastructure and source code will meet the requirements of all courses from the INA specialization, and thus can be used as the source of all teaching materials. Although current courses are thematically related and linked together, teaching materials do not have any common base. The management layer will be built on the basis of Jakarta EE as a front and back-end with a standard RDBMS and run on top of the **OpenShift** platform. The solution is going to be highly available, scalable, monitored and resistant to hardware and software failures. The **VMware vSphere** cluster as shown on Figure 3 will be the foundation of all of this of course. Remote laboratory stands and infrastructure has became permanent element of our teaching process. Even if the pandemic situation returns to "normal" state, presented educational infrastructure will be used during INA specialization courses. Without any problems we can offer courses to Erasmus students and they do not have to be in Poland at all. They can sit comfortably at their homes what has significant meaning considering possible traveling restrictions and quarantine procedures.

Acknowledgment

The authors would like to thank:

- PhD. Eng. Michał Morawski for NETLAB installation, administration, scripts and pods configuration,
- PhD. Eng. Raman Krasiukianis for pods configuration on NETLAB
- themselves for the rest of infrastructure installation, administration and development.

References

[1] Atlassian Solutions. https://www.atlassian.com/, 2021.

[2] Nafalski, A., Milosz, M., Considine, H., and Nedić, Z. Overseas Use of the Remote Laboratory NetLab. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 568–573. 2020. doi:10.1109/EDUCON45650.2020.9125230.

[3] Elder, M., Kitchener, J., and Topol, B. *Hybrid Cloud Apps with OpenShift and Kubernetes: Delivering Highly Available Applications and Services*. O'Reilly Media, 2021.

[4] Pittet, S. Continuous integration vs. continuous delivery vs. continuous deployment. https://www.atlassian.com/continuous-delivery/principles/continuous-integration-vs-delivery-vs-deployment, 2020.

[5] Payara Platform Documentation. https://www.payara.fish/learn/payara-platform-documentation/, 2021.

[6] Jakarta EE Platform. https://jakarta.ee/specifications/platform/9.1/jakarta-platform-spec-9.1.pdf, 2021.

[7] The PostgreSQL Global Development Group. *PostgreSQL 13.3 Documentation*, 2021.

[8] Sakimura, N., Bradley, J., Jones, M., de Medeiros, B., and Mortimore, C. Openid connect core 1.0. https://openid.net/developers/specs/, 2014.

[9] Hardt, D. The OAuth 2.0 Authorization Framework. RFC 6749, 2012. doi: 10.17487/RFC6749. URL `https://rfc-editor.org/rfc/rfc6749.txt`.