UNIVERSITY OF CRETE
DEPARTMENT OF COMPUTER SCIENCE
FACULTY OF SCIENCES AND ENGINEERING

CognitOS Classboard: a Multimodal Interaction Framework for Enhancing the Whiteboard of the Intelligent Classroom

by

George Nikitakis

MASTER'S THESIS

Heraklion, June 2020

UNIVERSITY OF CRETE

DEPARTMENT OF COMPUTER SCIENCE

CognitOS Classboard: a Multimodal Interaction Framework for Enhancing the Whiteboard of the Intelligent Classroom

by **George Nikitakis**

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science

APPROVED BY:

Author: George Nikitakis

Supervisor: Constantine Stephanidis, Professor, University of Crete

CONSTANTINOS STEPHANIDIS

Digitally signed by CONSTANTINOS STEPHANIDIS
Date: 2020.06.25 19:58:17 +03'00'

Committee Member: Georgios Papagiannakis, Associate Professor, University of Crete

GEORGIOS PAPAGIANNAKIS

Digitally signed by GEORGIOS PAPAGIANNAKIS
Date: 2020.06.25 19:58:16 +03'00'

Committee Member: Dimitris Grammenos, Principal Researcher, FORTH-ICS

DIMITRIOS-STAVROS GRAMMENOS

Digitally signed by DIMITRIOS-STAVROS GRAMMENOS
Date: 2020.06.25 20:27:12 +03'00'

Director of Graduate Studies: Antonis Argyros, Professor, University of Crete

Digitally signed by ANTONIOS ARGYROS
Date: 2020.06.25 20:35:10 +03'00'

Heraklion, June 2020

This thesis is dedicated to my family…
to my father Stelios, my mother Eftyhia,
my sister Maria, and my brother Andreas
… who have always supported me

Acknowledgments

"As you set out for Ithaka
hope the voyage is a long one,
full of adventure, full of discovery."

C.P. Cavafy

During this long… journey there are some people that continuously supported, influenced, and helped me, and I would like to thank all of them…

First of all, I would like to thank my thesis supervisor, Professor Constatine Stephanidis, for giving me the opportunity to be a member of the HCI laboratory, and for his support, assistance and guidance throughout my Master studies. I am also grateful to my advisory committee members, Associate Professor Georgios Papagiannakis and Principal researcher Dimitris Grammenos, who helped me improve my work with their valuable feedback and critical thinking. I would also like to acknowledge Dr. Margherita Antona for her valuable feedback.

Next, I need to thank the Department of Computer Science (CSD) of the University of Crete, and the Human Computer Interaction (HCI) Laboratory of the Institute of Computer Science (ICS) of the Foundation for Research and Technology - Hellas (FORTH) for equipping me with the means and creating a highly motivational working environment to achieve my research goals.

Moreover, I owe my appreciation to my colleagues Dr. Asterios Leonidis and Dr. Maria Korozi; first of all, for believing in me…, for their continuous

guidance and valuable advice, which contributed the most to shaping this work into its current state. I would also like to thank Natasa Ntagianta for her contribution in the entire Design Process of this work, Despoina Gavgiotaki for designing and developing some applications that were employed by the proposed system, and Eleni Stephanidi for sharing with me her knowledge regarding the Intelligent Classroom artifacts. Special thanks to all the users that participated in the system's evaluation process, and to Titos Michelakis for recording and editing the footage of the system's promotion video.

My best thanks to my friends Dimitris Arampatzis, Vasilis Glampedakis, Nikos Liapakis, Michalis Loukakis, David Parastatidis, George Psaromiligkos, Kostas Rousochatzakis, Pavlos Tzourmpakis, Nikos Papadakis, Stephanos Christophorakis, Tasos Georgakakis, and Jonny Mouzakitis who supported me directly or indirectly and inspired me during my studies and generally in my life. Additionally, I would like to thank my friends from FORTH Dimitris Arampatzis, Iraklis Bekiaris, Antonis Dimopoulos, Maria Doulgeraki, Nantia Manoli, Andreas Mixelakis, Natasa Ntagianta, Vangelis Poutouris, Anastasia Rigaki, Evropi Stefanidi, Eirini Sykianaki annnd Elena Tsolakou for supporting me, and making FORTH a fun and interesting place.

Furthermore, I acknowledge with deep sense of reverence my gratitude to my spiritual father Metropolitan of Lampe, Syvritos and Sfakia Irenaios for his limitless support and guidance since I was a child. I also have no valuable words to express my thanks to Archimandrite Nikiforos Kounalis, Preacher of the Holy Archdiocese of Crete and Principal of the Regional Ecclesiastical Student Dormitories for the accommodation in the dorms and his effort to make me feel like home. Additionally, I would like to thank each and every one of the students living in the dorms for supporting and encouraging me during this period, especially my roommate Nikos Daniolos whom I woke up when returning home after midnight during the last months of my studies,

thanks again Nikos and Sotiris Statheas who always saved food for me, and Giorgos Sofianos who reminded me what actually matters in life.

I take this opportunity to record my sincere thanks to my undergraduate supervisor Professor Antonis Argyros for his guidance through my undergraduate studies, to my supervisor Associate Professor Yannis Tzitzikas for his support and his advice during my undergraduate thesis and first semesters of my master studies, and also Panagiotis Papadakos for his assistance and guidance during my undergraduate thesis, for inspiring me to improve my skills and of course for teaching me how to properly use git.

Finally, I am more than grateful to my parents Stelios and Eftyhia, my sister Maria and my brother Andreas for their endless love, support and encouragement through my studies and my life in general. Additionally, I am deeply indebted to my father's brother my uncle George Nikitakis. I would not have made it this far without all of you. Thank you!

As a Cretan guy…

"A big thank you to my friends, my family,
and to those who supported me and were beside me"

George Nikitakis

Ευχαριστίες

"Σα βγεις στον πηγαιμό για την Ιθάκη,
να εύχεσαι να 'ναι μακρύς ο δρόμος,
γεμάτος περιπέτειες, γεμάτος γνώσεις."

Κ.Π. Καβάφης

Κατά τη διάρκεια αυτού του μεγάλου… ταξιδιού υπήρξαν πολλοί άνθρωποι που με υποστήριξαν, με επηρέασαν και με βοήθησαν, και θα ήθελα να τους ευχαριστήσω όλους…

Καταρχάς, θα ήθελα να ευχαριστήσω τον επόπτη της μεταπτυχιακής μου εργασίας, καθηγητή Κωνσταντίνο Στεφανίδη, που μου έδωσε την ευκαιρία να γίνω μέλος του εργαστηρίου Αλληλεπίδρασης Ανθρώπου Υπολογιστή και ταυτόχρονα για την υποστήριξη, τη βοήθεια και την καθοδήγηση του κατά τη διάρκεια των μεταπτυχιακών μου σπουδών. Είμαι επίσης ευγνώμων στα μέλη της εξεταστικής επιτροπής, τον αναπληρωτή καθηγητή Γεώργιο Παπαγιαννάκη και τον κύριο ερευνητή Δημήτρη Γραμμένο, που με βοήθησαν να βελτιώσω τη δουλειά μου με τα πολύτιμα σχόλια και την κριτική σκέψη τους. Θα ήθελα επίσης να ευχαριστήσω τη Δρ. Μαργαρίτα Αντόνα για τα πολύτιμα σχόλια της.

Στη συνέχεια, νιώθω την ανάγκη να ευχαριστήσω το Τμήμα Επιστήμης Υπολογιστών του Πανεπιστημίου Κρήτης και το Εργαστήριο Αλληλεπίδρασης Ανθρώπου Υπολογιστή του Ινστιτούτου Πληροφορικής του Ιδρύματος Τεχνολογίας και Έρευνας (ΙΤΕ) για την παροχή εξοπλισμού και τη δημιουργία ενός κατάλληλου περιβάλλοντος εργασίας για την επίτευξη των ερευνητικών μου στόχων.

Επιπλέον, θέλω να εκφράσω την εκτίμηση μου στους συναδέλφους μου Δρ. Αστέριο Λεωνίδη και τη Δρ. Μαρία Κορόζη, πρώτα από όλα γιατί πίστεψαν σε μένα…, για τη συνεχή καθοδήγηση και τις πολύτιμες συμβουλές τους, οι οποίες συνέβαλαν τα μέγιστα στη διαμόρφωση αυτής της εργασίας στην παρούσα της κατάσταση. Θα ήθελα επίσης να ευχαριστήσω την Νατάσα Νταγιαντά για τη συμβολή της σε ολόκληρη τη διαδικασία σχεδίασης της εργασίας, τη Δέσποινα Γαυγιωτάκη για το σχεδιασμό και την ανάπτυξη ορισμένων εφαρμογών που χρησιμοποιήθηκαν από το προτεινόμενο σύστημα, και την Ελένη Στεφανίδη που μοιράστηκε μαζί μου τις γνώσεις της σχετικά με τα έξυπνα αντικείμενα της τάξης. Θα ήθελα να ευχαριστήσω ιδιαίτερα, όλους τους χρήστες που συμμετείχαν στη διαδικασία αξιολόγησης του συστήματος, και τον Τίτο Μιχελάκη για την καταγραφή και επεξεργασία του βίντεο για την προώθηση του συστήματος.

Ευχαριστώ θερμά τους φίλους μου Δημήτρη Αραμπατζή, Βασίλη Γλαμπεδάκη, Νίκο Λιαπάκη, Μιχάλη Λουκάκη, Δαβίδ Παραστατίδη, Γιώργο Ψαρομήλιγκο, Κώστα Ρουσοχατζάκη, Παύλο Τζουρμπάκη, Νίκο Παπαδάκη, Στέφανο Χριστοφοράκη, Τάσο Γεωργακάκη, και Γιάννη Μουζακίτη για την άμεση και έμμεση στήριξη, για τα κίνητρα και την έμπνευση που μου πρόσφεραν τόσο κατά τη διάρκεια των σπουδών μου αλλά και γενικότερα στη ζωή μου. Επιπλέον, θα ήθελα να ευχαριστήσω την παρέα μου από το ΙΤΕ Δημήτρη Αραμπατζή, Ηρακλή Μπεκιάρη, Αντώνη Δημόπουλο, Μαρία Δουλγεράκη, Νάντια Μανώλη, Αντρέα Μιχελάκη, Νατάσα Νταγιαντά, Βαγγέλη Πουτούρη, Αναστασία Ρηγάκη, Ευρώπη Στεφανίδη, Ειρήνη Συκιανάκη καιιι Έλενα Τσολάκου οι οποίοι με υποστηρίζουν, και κάνουν το ΙΤΕ ένα διασκεδαστικό και ενδιαφέρον μέρος.

Επιπλέον, αναγνωρίζω με βαθιά αίσθηση σεβασμού την ευγνωμοσύνη μου στον πνευματικό μου τον Σεβασμιώτατο Μητροπολίτη Λάμπης, Συβρίτου και Σφακίων κ.κ. Ειρηναίο για την απεριόριστη υποστήριξη και καθοδήγηση του από όταν ήμουν μικρό παιδί. Δεν έχω επίσης λόγια για να εκφράσω τις ευχαριστίες μου στον Αρχιμ. Νικηφόρο Κουνάλη, Ιεροκήρυκα της Ι.Α.Κ. και Διευθυντή της Περιφερειακής Εκκλησιαστικής Εστίας για τη διαμονή στους κοιτώνες της φοιτητικής εστίας και την προσπάθεια του να με κάνει να νιώσω σαν το σπίτι μου. Επιπρόσθετα, θα ήθελα να

ευχαριστήσω όλους τους φοιτητές που διαμένουν στη φοιτητική εστία οι οποίοι με στήριξαν και με ενθάρρυναν κατά τη διάρκεια αυτής της περιόδου, ειδικά το συγκάτοικο μου Νίκο Δανιόλο τον οποίο ξυπνούσα κάθε βράδυ όταν επέστρεφα στην εστία, μετά τα μεσάνυχτα, τους τελευταίους μήνες των σπουδών μου, ευχαριστώ και πάλι το Νίκο και το Σωτήρη Σταθέα που πάντα μου κρατούσαν φαγητό, και το Γιώργο Σοφιανό που μου θύμισε τι πραγματικά έχει αξία στη ζωή.

Παίρνω αυτή την ευκαιρία για να αναφέρω ένα μεγάλο ευχαριστώ στον προπτυχιακό μου σύμβουλο καθηγητή κύριο Αντώνη Αργυρό για την καθοδήγηση του κατά τη διάρκεια των προπτυχιακων μου σπουδών, στο σύμβουλο μου καθηγητή κύριο Γιάννη Τζίτζικα για την υποστήριξη και τις συμβουλές του κατά τη διάρκεια εκπόνησης της προ-πτυχιακής μου εργασίας καθώς και στα πρώτα εξάμηνα των μεταπτυχιακών μου σπουδών. Όπως επίσης, θα ήθελα να ευχαριστήσω τον Παναγιώτη Παπαδάκο για τη βοήθεια και την καθοδήγηση του στην προπτυχιακή μου εργασία, για την έμπνευση που μου έδινε να βελτιώσω τις δεξιότητες μου και φυσικά γιατί με δίδαξε να χρησιμοποιώ σωστά το git.

Τέλος, είμαι ευγνώμων στους γονείς μου Στέλιο και Ευτυχία, την αδερφή μου Μαρία και τον αδερφό μου Αντρέα για την ατελείωτη αγάπη, υποστήριξη και το θάρρος που μου έδιναν κατά τη διάρκεια των σπουδών μου αλλά και στη ζωή μου γενικά. Επιπλέον, χρωστάω πολλά στον αδερφό του πατέρα μου το Θείο μου Γιώργο Νικητάκη. Δε θα είχα φτάσει τόσο μακριά χωρίς εσάς. Ευχαριστώ!

Σαν Κρητικός…

"Ένα μεγάλο ευχαριστώ σε φίλους, συγγενείς μου,
και σ' όσους με στηρίξατε και ήσασταν μαζί μου"

Γιώργος Νικητάκης

Abstract

Technology has already permeated education at all levels, ranging from pre-school to higher education. When used appropriately, Information and Communication Technologies (ICTs) can transform learning into an engaging and interactive process and enhance teaching with the use of educator-oriented tools. Nowadays, Interactive Whiteboards (IWbs) stand out as a widespread technological equipment that has already permeated classrooms, with their usage extending from pre-school to university. Many advantages exist in integrating IWBs into the classroom environment. They are considered as a great tool to improve the educational process, while their acceptance is constantly growing; that is because they enable educators to express their creativity, but equally important, because they encourage students' active participation.

This work presents the CognitOS Classboard, an educator- and student-oriented framework, employed on the "Intelligent Classroom Board" - a wall-to-wall projected Interactive board - which takes advantage of the facilities offered by the intelligent environment and the amenities stemming from its large dimensions (wall-size), in order to enhance educational activities and provide new educator- and student- oriented experiences.

From the students' perspective, the CognitOS Classboard, apart from offering access to useful educational applications, provides mechanisms capable of transforming the classroom into an immersive environment on demand. Such a facility can enable interaction with the real world in ways that were not possible before (e.g. immerse students into a rainforest). Additionally, taking advantage of its multi-touch functionality, the proposed system supports student collaboration in order to enhance the learning process.

From the educators' perspective, the CognitOS Classboard aims to support lecturing. In particular, the main goal is to provide a unified working environment for the large interactive board, offering a straightforward 'natural' interaction paradigm, which allows educators to orchestrate their teaching assets in an optimal way (e.g. use different workspaces to place different types of applications) and present information-rich material to the students. Additionally, it aims to support course preparation before class, by equipping educators with user-friendly tools that allow them to initialize the contents of the board with the appropriate material in advance.

Aiming to blend seamlessly in the classroom environment and in keeping with the already existing standards and processes of teaching / lecturing, the CognitOS Classboard enhances well-established interaction metaphors (e.g. writing on the digital board is similar to writing on a plain whiteboard) and supports multimodal interaction through: i) touch on the wall surface with the finger or a special marker, ii) a remote presenter, iii) voice commands, iv) sophisticated techniques that take advantage of the Intelligent Classroom's Ambient Intelligence facilities (e.g. through user position tracking, an application window can tail the educator while moving around), and v) the controller application for handheld devices (tablet).

This thesis presents the CognitOS Classboard, describes the design process and its functionality, elaborates on the implementation details and discusses the results of a user-based evaluation study in the context of an Intelligent (University) Classroom.

Keywords: Class Board, Interactive Whiteboard, Interactive Wall, Intelligent Classroom, Ambient Intelligence

Περίληψη

Είναι γνωστό ότι η τεχνολογία έχει ήδη διεισδύσει σε όλες τις βαθμίδες της εκπαίδευσης, από την προσχολική έως και την τριτοβάθμια. Όταν χρησιμοποιούνται κατάλληλα, οι Τεχνολογίες Πληροφοριών και Επικοινωνίας (ΤΠΕ) μπορούν να μετατρέψουν τη μάθηση σε μια ελκυστική και διαδραστική διαδικασία, και να βελτιώσουν την ποιότητα της εκπαίδευσης εξοπλίζοντας τους διδάσκοντες με κατάλληλα εργαλεία. Οι αλληλεπιδραστικοί πίνακες ξεχωρίζουν ως ένας διαδεδομένος τεχνολογικός εξοπλισμός που έχει επικρατήσει στις σχολικές αίθουσες, με τη χρήση τους να εκτείνεται από το νηπιαγωγείο έως το πανεπιστήμιο. Υπάρχουν πολλά πλεονεκτήματα που απορρέουν από τη χρήση αλληλεπιδραστικών πινάκων στο περιβάλλον της τάξης, καθώς θεωρούνται ως εξαιρετικά εργαλεία για τη βελτίωση της εκπαιδευτικής διαδικασίας, ενώ η αποδοχή τους συνεχώς αυξάνεται. Αυτό συμβαίνει επειδή επιτρέπουν στους εκπαιδευτικούς να εκφράσουν τη δημιουργικότητα τους, αλλά εξίσου σημαντικό, επειδή ενισχύουν τη συμμετοχή των μαθητών.

Αυτή η εργασία παρουσιάζει το CognitOS Classboard, ένα σύστημα το οποίο αφορά τόσο στον καθηγητή όσο και στους μαθητές, και βρίσκεται εγκατεστημένο στον διαδραστικό πίνακα του χώρου προσομοίωσης της «Έξυπνης Τάξης». Πρόκειται για έναν «έξυπνο» αλληλεπιδραστικό πίνακα που προβάλλεται σε τοίχους της αίθουσας διδασκαλίας και εκμεταλλεύεται τις «έξυπνες» υπηρεσίες που προσφέρονται από το περιβάλλον της «Έξυπνης Τάξης» και τις δυνατότητες που απορρέουν από τις μεγάλες διαστάσεις του (μέγεθος τοίχου), προκειμένου να προάγει τις εκπαιδευτικές δραστηριότητες και να παρέχει εμπειρίες προσωποποιημένες στον εκάστοτε καθηγητή και τους μαθητές του.

Από την οπτική γωνία των μαθητών, εκτός από την παροχή πρόσβασης σε χρήσιμες εκπαιδευτικές εφαρμογές, το CognitOS Classboard προσφέρει μηχανισμούς ικανούς να μετατρέψουν την τάξη σε ένα εμβυθιστικό περιβάλλον (immersive environment). Μια τέτοια λειτουργία επιτρέπει την αλληλεπίδραση με τον πραγματικό κόσμο με

τρόπους που δε θα ήταν εφικτοί διαφορετικά (π.χ. να εμβυθίσει τους μαθητές μέσα σε ένα τροπικό δάσος). Επιπλέον, εκμεταλλευόμενο τη λειτουργικότητα πολλαπλής αφής, το προτεινόμενο σύστημα υποστηρίζει τη συνεργασία των μαθητών με σκοπό τη βελτίωση της εκπαιδευτικής διαδικασίας.

Από εκπαιδευτικής σκοπιάς, το CognitOS Classboard στοχεύει στην υποστήριξη της διδασκαλίας. Ειδικότερα, ο κύριος στόχος είναι να παρέχει ένα ενοποιημένο περιβάλλον εργασίας για τον πίνακα, στο οποίο η αλληλεπίδραση γίνεται με απλό 'φυσικό' τρόπο, επιτρέποντας στους εκπαιδευτικούς να οργανώσουν κατάλληλα τα ψηφιακά τους εργαλεία (π.χ. με τη χρήση διαφορετικών επιφανειών εργασίας για την τοποθέτηση διαφορετικών τύπων εφαρμογών) προσφέροντας πλούσιο ψηφιακό υλικό στους μαθητές. Επιπλέον, στοχεύει στην υποστήριξη της προετοιμασίας των μαθημάτων, εξοπλίζοντας τους εκπαιδευτικούς με φιλικά εργαλεία, που επιτρέπουν την αρχικοποίηση του πίνακα με το κατάλληλο υλικό πριν την έναρξη της διδασκαλίας.

Έχοντας ως στόχο να ενσωματωθεί με μη-επεμβατικό τρόπο στο περιβάλλον της τάξης και να εναρμονιστεί με τους ήδη καθιερωμένους τρόπους διδασκαλίας, το CognitOS Classboard εμπλουτίζει παραδοσιακούς τρόπους αλληλεπίδρασης (π.χ. η γραφή πάνω στον ψηφιακό πίνακα είναι παρόμοια με τη γραφή σε έναν απλό πίνακα), ενώ υποστηρίζει πολυτροπική αλληλεπίδραση μέσω των εγκαταστάσεων της «Έξυπνης τάξης». Η αλληλεπίδραση επιτυγχάνεται μέσω: (α) της αφής ή με τη χρήση ενός ειδικού «στυλό» στην επιφάνεια του «Έξυπνου Πίνακα», (β) ενός χειριστηρίου παρουσιάσεων (remote presenter), (γ) φωνητικών εντολών, (δ) εξεζητημένων τεχνικών που βασίζονται στις τεχνολογίες Διάχυτης Νοημοσύνης που βρίσκονται στο περιβάλλον της «Έξυπνης Τάξης» (π.χ. κατάλληλη ανανέωση της θέσης των παραθύρων ώστε να βρίσκονται κοντά στη φυσική θέση του χρήστη στον χώρο για να διευκολύνεται η πρόσβαση σε αυτά) και (ε) μέσω της εφαρμογής διαχείρισης για φορητές συσκευές (tablet).

Αυτή η εργασία παρουσιάζει το CognitOS Classboard, περιγράφει τη διαδικασία σχεδίασης και τη λειτουργικότητά του, αναλύει λεπτομέρειες σχετικά με την υλοποίησή του, και καταλήγει παρουσιάζοντας τα αποτελέσματα ενός πειράματος

αξιολόγησης, το οποίο έγινε με τη συμμετοχή χρηστών στο πλαίσιο μιας «Έξυπνης (Πανεπιστημιακής) Τάξης».

Λέξεις κλειδιά: Πίνακας διδασκαλίας, Διαδραστικός Πίνακας, Διαδραστικός Τοίχος, Έξυπνη Τάξη, Διάχυτη Νοημοσύνη

Contents

Acknowledgments vii
Ευχαριστίες xi
Abstract xv
Περίληψη xvii
Contents xxi
List of Figures xxvii
List of Tables xxxi
Introduction 1
1.1 Interactive technology in the classroom 1
1.2 Interactive Whiteboards and Education 4
1.3 Ambient Intelligence and Intelligent Classrooms 5
1.4 Objectives of the CognitOS Classboard 7
1.5 Thesis Outline 9
Related Work 11
2.1 Hardware-specific solutions 11
2.2 Hardware-independent solutions 15
2.3 Discussion 23
Design Process 25
3.1 Methodology 25
3.2 Motivating Scenarios 27
3.2.1 Teaching a University Class 28
3.2.2 Teaching a History Class 29
3.3.3 Meeting Room Presentation 30
3.3 Requirements 32
3.3.1 Functional Requirements 32
3.3.2 Non-Functional Requirements 35

The Classroom behind CognitOS Classboard 37
4.1 Software Infrastructure 37
4.1.1 AmI Solertis 38
4.1.2 ClassMATE 39
4.1.3 LECTOR 40
4.1.4 CognitOS 42
4.2 Artefacts of the Intelligent Classroom 45
4.2.1 Intelligent Student Desk 46
4.2.2 Interactive Classroom Board 46
4.2.3 Educator's Workstation 47
The CognitOS Classboard 49
5.1 Design Challenges 49
5.2 Fundamental Components 51
5.3 CognitOS Classboard Functionality 52
5.3.1 CognitOS Classboard 52
General Functionality 53
Immersive Experiences & X-Reality Technologies 59
5.3.2 Board Component 61
5.3.3 Workspace Component 64
5.3.4 Window Component 65
5.4 Interaction Paradigm 67
5.4.1 On-surface Interaction 68
5.4.2 Above-surface Interaction 69
5.4.3 Remote Interaction from Desktop & Mobile platforms 69
Desktop Application 70
Tablet Application 71
Smartphone Application 72
5.5 Applications Suite 73
5.5.1 Educational Applications 73
5.5.2 Utilities 74

CognitOS Classboard Implementation 77
6.1 CognitOS Classboard Core 79
6.1.1 Classboard controller 79
6.1.2 Logical board manager 81
6.1.3 UI Components Usher 81
6.1.4 State manager 89
Applications state 90
Windows state 91
Workspaces state 93
Boards state 94
CognitOS Classboard state 94
6.1.5 Interaction with CognitOS Classboard 95
6.1.6 Reachability recognition 106
6.1.7 Classroom's bridge 107
6.1.8 CognitOS Classboard as a service 108
System 110
Board 111
Workspace 116
Window 124
Applications 132
6.2 User Interface Framework 134
6.2.1 Implementation of fundamental UI components 134
6.2.2 Decomposition of Classboard's UI layers 135
6.3 CognitOS Classboard Utilities 140
6.3.1 State monitoring 140
State monitoring backend 141
6.3.2 Remote controllers' manager 142
6.3.3 Board Configurator service 142
6.4 Classroom services library 142
6.4.1 Environment's state 143

6.4.2 Interaction Manager 144
6.4.3 User Profile 146
User recognition service 147
6.3.4 Datastore service 147
6.3.4 Event federator 148
6.4 The Applications Suite Mechanics 149
6.4.1 Incorporating a new educational application 149
6.4.2 Applications Suite Manager 152
Applications Suite Manager Backend 153
Applications Suite Manager Frontend 153
Evaluation 155
7.1 Cognitive Walkthrough 155
7.1.1 Process 156
7.1.2 Results 156
7.2 User Based Evaluation 157
7.2.1 Research Questions 158
7.2.2 Participants 159
7.2.3 Data Collection 161
Pre-evaluation Questionnaire 161
Observation Grid 162
Post-evaluation Questionnaires 162
7.2.4 Procedure 163
Preparation 163
Introduction 163
Running the test 164
Debriefing 165
7.2.5 Results 165
Research Question 1: Is the CognitOS Classboard usable? 165
Research Question 2: Is the integrated functionality useful? 168
Research Question 3: Do users exhibit interaction related issues? 170

Conclusion and Future Work 173
Bibliography 177
Appendix A 191
User-Based Evaluation Task Scenarios 191

List of Figures

Figure 1: Could this be how the classroom of the future might look? [9]. 3
Figure 2: Education room illustration enhanced with EPSON technology [38]. 11
Figure 3: ViewBoard doubles the learning with two independent pens [39]. 12
Figure 4: EZWrite is an easy and fun annotation solution [40] 13
Figure 5: MagicIWB, Samsung's powerful, intuitive Interactive Whiteboard software [41]. 14
Figure 6: Oktopus software supporting collaboration to enhance learning [42]. 15
Figure 7; Adaptation of a multimedia application to meet the needs of different artifacts [44]. 16
Figure 8: Intellichalk deployed in an intelligent classroom [46]. 17
Figure 9: Smart learning suite software [47]. 17
Figure 10: ActivInspire on the Promethean ActivPanel [49]. 18
Figure 11: Canvus collaboration software equipped in an intelligent classroom [51]. 19
Figure 12: Interactive playground..... 20
Figure 13. OneScreen Annotate interactive whiteboard software [54]..... 21
Figure 14: Annotate engage your entire class [55]. 22
Figure 15: Design Thinking Process [58]. 26
Figure 16: Designing lo-fidelity prototypes on the Interactive Classroom Board. 27
Figure 17: Hi-fidelity prototypes of the CognitOS Classboard. 27
Figure 18: The AmI-Solertis Hybrid Communication Protocol [64] 38
Figure 19: The desktop of CognitOS running on the augmented school desk.42

Figure 20: A notification appears trying to encourage a student during exercise solving..... 43
Figure 21: The Intelligent Classroom of FORTH-ICS [33]. 45
Figure 22: a. Representation of the CognitOS Classboard components hierarchy, and b. Visualization of a sample system instantiation..... 51
Figure 23: Representation of Alternative Workspaces. 52
Figure 24: a. Follow-me, and b. Summon features. 54
Figure 25: Summoning an active window inside a workspace. 54
Figure 26: Draw attention feature. 55
Figure 27: Annotation functionality. 55
Figure 28: Capture functionality. 56
Figure 29: The board manager..... 58
Figure 30: 3D representation of the CognitOS Classboard immersing students into the Dordogne caves. 60
Figure 31: Paging component. 61
Figure 32: Board-specific menu. 63
Figure 33: Workspace-specific Menu. 64
Figure 34: Reachable interactive components. 65
Figure 35: Show / hide frame feature. 66
Figure 36: Window-specific Menu (Window residing on the Board). 67
Figure 37:Window-specific Menu (Window residing in a Workspace). 67
Figure 38: 3D representation of the CognitOS Classboard displaying educational applications and a picture taken from in the actual Intelligent Classroom setup. 68
Figure 39: The desktop application. 71
Figure 40: The tablet application running on the educator's workstation. ... 72
Figure 41: The smartphone application..... 72
Figure 42: (a) Presentation viewer, (b) Image viewer, and (c) Video player.. 74
Figure 43: a. Calendar application, b. Submissions application, c. Clock application analogue mode, d. Clock application stopwatch mode. 75

Figure 44: The four basic modules that constitue CognitOS Classboard framework and the Classroom's enviroment services module. 78
Figure 45: State related services and how they communicate. 90
Figure 46: Describes in (a) a pan gesture [109], and in (b) a press gesture [110]. 97
Figure 47: Describes a rotate gesture [111]..... 98
Figure 48: Describes a pinch gesture [112]..... 99
Figure 49: Describes in (a) swipe gesture [113], and in (b) tap gesture [114].100
Figure 50: Visualizes the shapes that can be recognized. 101
Figure 51: Classboard's user interface layers. 136
Figure 52: Events travel from window to target element and back to window. 138
Figure 53: Participants' experience over lectures / presentations. 160
Figure 54: Use of board during lecture. 160
Figure 55: Use of projector during lecture. 161
Figure 56: Introduction stage of evaluation. 164
Figure 57: Running the test. 165
Figure 58: Errors and hints distribution per user. 167
Figure 59: Curved grading scale for SUS [127]. 168
Figure 60: Overall SUS score per user.....168
Figure 61: A snapshot of CognitOS Classboard. 174
Figure 62: Design Thinking Process. 191
Figure 63: Empathic modelling.....191
Figure 64: Prototyping. 192
Figure 65: Board's state after the completion of Task 3. 193
Figure 66: Login box prototype. 194
Figure 67: The captured area. 195

List of Tables

Table 1: Comparison Matrix of Board Software..... 24
Table 2: For each menu's mode which components are visible. 57
Table 3: Functionality of CognitOS Classboard Boards. 62
Table 4: Functionality of CognitOS Classboard Workspaces..... 63
Table 5: Functionality of CognitOS Classboard Windows..... 66
Table 6: A sample of the defined shortcuts. 104
Table 7: CognitOS Classboard API functions. 109
Table 8: The frequency of the errors made by the users. 166

Chapter 1

Introduction

1.1 Interactive technology in the classroom

In the past, researchers have studied the efficacy of technology in the domain of education; when used appropriately, Information and Communication Technologies (ICTs) can transform learning into an engaging and interactive process and enhance teaching with the use of educator-oriented tools [1].

Technology has already permeated education in all levels, ranging from pre-school to higher education. A notable example is the emergence of Digital Storytelling [2], [3] is a powerful tool aiming to capture the attention of learners and especially (but not exclusively) children. It is based on the idea of creating a traditional story enhanced with multimedia, which when selected carefully promote active learning [4] by putting the learners in control and enabling their active participation. Interestingly, as soon as it appeared, Digital Storytelling permeated pre-school education, was well received by educators and became integrated into a variety of educational activities [5].

Apart from Digital Storytelling, another popular trend to learning is Stem Education, which integrates the areas of science, technology, engineering and mathematics. Information and Communication Technologies are acknowledged as very important tools for STEM Education as well [6], since they increase the opportunities for teaching and learning of science,

technology, engineering, and mathematics. With ICTs, educators and students have access to the Internet, audio-visual classrooms, social networking, and many more facilities. The aforementioned benefits explain the wide numbers of research that focus on educational technologies in STEM education providing innovative applications that support educators and students [7].

Regarding the incorporation of technology into higher education, [8] reports that it:

- permits students to access learning opportunities apart from the traditional barriers of time and place
- enables access to learning opportunities outside of formal higher education institutions, such as at their workplace or in community settings
- helps students to access high-quality learning resources, regardless of their institution's geographical location or funding
- supports enhanced learning experiences through blended learning models
- supports students in their learning based on individual academic and non-academic needs through personalization
- ensures that students with disabilities participate in and benefit from educational programs and activities.

Additionally, [8] describes the benefits of teaching with the help of ICTs. In particular, it mentions that through technology-enabled teaching instructors:

- can use data gathered about student learning to provide targeted interventions and tailored feedback
- can use student learning data to evaluate the efficacy of new teaching practices or new technologies
- are equipped with the means of creating active learning environments that connect students with content in different ways
- can use tools to provide personalized and connected experiences to all students

- can use tools to provide high-quality resources to students at a lower cost

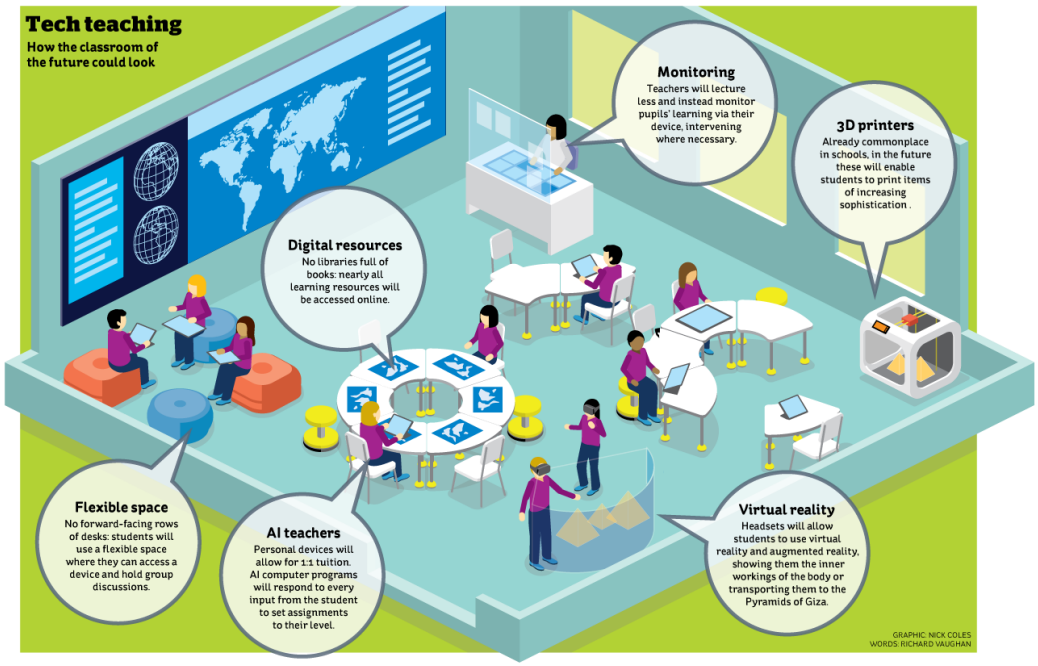


Figure 1: Could this be how the classroom of the future might look? [9].

More recently, industry [10], [11] and online communities [12] have discussed the hardware infrastructure that future classrooms should incorporate, including:

- **Interactive Boards** or **Interactive Walls**. A highly interactive wall will deliver key learning points, keep children engaged and allow for collaboration.
- **Digital Student Devices** (i.e. smartphones, tablets, ebooks) and **Interactive Desks**. Apart from their handheld devices, students of the future will benefit from touch-screen technology at their own desks.
- **Charging Solutions**. Mass deployment of tablets, and similar devices in schools, requires sophisticated charging solutions that ensure that each device is neatly stored, securely contained and ready for the day's activities.

- **Visualizers** (Document Cameras), enable teachers and students to instantly stream vivid images of any object—living or inanimate, still or moving, flat or 3D—in real time to a large audience.
- **Lecture Recording Cameras**
- **3D Printers** that permit to literally bring theoretical, abstract ideas to life in tangible form, making learning more effective and fun.
- **X-Reality technologies** that motivate children to learn by cultivating their imagination and feeding their innate curiosity.

1.2 Interactive Whiteboards and Education

The term Interactive Whiteboard (IWB) (also known as electronic or digital board) refers to a large, touch-sensitive digital board designed to replace traditional blackboards (chalkboards) and whiteboards [13], [14]. They were initially introduced in 1991, aiming to enhance teaching by (i) enabling educators to display anything to the entire classroom, (ii) providing access to electronic learning resources [15], and (iii) exploiting the benefits of using audio-visual material during a lecture. From a technological perspective, IWBs are projector-based systems fixed in a permanent position, or mobile to enable relocation. Recently, large-format touchscreen displays, also called interactive flat panels (IFPs) came to the spotlight as a cheaper alternative to the projector-based solution. However, if the requirement is to create a large display area (e.g. 5760 x 1200), combining multiple projectors seems more appropriate, since in that case the bezels of the flat panels will hinder the continuity of the presented material and affect touch interaction.

Nowadays, Interactive Whiteboards stand out as the most widespread technological equipment that has already permeated the domain of education. Their usage ranges from pre-school to university; many commercial interactive whiteboards are already in the wild, while there are several research approaches aiming to create custom technologically enhanced interactive boards [16], [17]. Interestingly, according to [18] the education

sector is expected to lead the IWB market, which is expected to grow at a Compound Annual Growth Rate (CAGR) of 3.69% from 2018 to 2023.

There are many advantages of integrating IWBs into the classroom environment. They are considered as a great tool to improve the educational process, while their acceptance is constantly growing; that is because they enable educators to express their creativity, while at the same time generate a higher motivation for students [19]. Particularly, the affordances of large interactive classroom boards include visibility, stability, direct manipulation, multimodality, and re-usability, offering support for cumulative, collaborative, and recursive learning [20]. Additionally, the research reported in [21] revealed that the contribution of IWBs to students' engagement during a lecture is undeniable, while it highlights that IWBs can contribute to the development of 21st century skills (i.e. Critical thinking, Creativity, Collaboration) in students. Finally, according to [20] interactive boards are the means for initiating dialogues, which have the potential to evolve and revolve around digital content created by both educators and students, such as collaborative exercises or a collaborative list of ideas about a topic of interest during a lecture. Hence, as a public display, a large board and its interactive tools unveil new opportunities for expressing publicly and evaluating ideas, by actively collaborating over a single display. Additionally, by displaying appropriate multimedia content, the explanation and therefore the comprehension of challenging concepts is inherently supported.

1.3 Ambient Intelligence and Intelligent Classrooms

The term Ambient Intelligence (AmI) refers to "*an emerging discipline that brings intelligence to our everyday environments and makes those environments sensitive to us*" [22]. Additionally, Aarts and Wichert [23] refer to AmI as "*sensitive, adaptive electronic environments that respond to the actions of persons and objects and cater for their needs*".

The emergence of the AmI paradigm lead to the appearance of user-centered intelligent environments that have the ability to actively support their users, sensibly and proactively in their everyday lives, by anticipating their needs [24], [25]. In the domain of education, Ambient Intelligence can significantly improve learning and overall student performance, but also support educators not only during the actual lecture, but also in their various educational tasks, such as preparing for a lecture, monitoring student progress and reviewing their performance [26]. In particular, Intelligent Classrooms [27] offer students and educators rich opportunities such as access to advanced X-Reality (i.e. augmented, virtual, and mixed reality) technologies, virtual agents or robots as personal assistants, and real time monitoring of student behavior in order to initiate interventions that help individuals in need [28].

One of the first projects towards the realization of an Intelligent Classroom environment [26], managed to capture audio, video, slides, and handwritten annotations during live lectures and make the recordings available to students. Additionally, it tried to facilitate the presentation process by automating several routines (e.g. adjust the lighting, set specific projector in motion, activate screens or displays). A similar approach is described in [29], where a number of embedded sensors and actuators were employed to control a number devices like lighting, air conditioning, presentation devices, and monitors. Apart from managing the conditions of the environment, the work in [30], exploits Radio Frequency IDs (RFID) readers in order to track students and educators within the classroom and automatically infer class attendance.

More recent approaches focus on the definition of efficient delivery models for smart classrooms that take advantage of ICT technologies (e.g. interactive smartboards, classroom control centers, mobile computing) [31] and the employment of multiagent technologies to define the intelligent capabilities of the included components and their interactions in a scalable and flexible way [16].

According to [32], the Intelligent Classroom requires a framework that satisfies five requirements, namely unobtrusive hardware, seamless and wireless

communications, intuitive and friendly user interface, safe and trusting environment to reassure learners, and dynamically distributed network to sustain all the other requirements.

The Intelligent Classroom of ICS-FORTH (http://ami.ics.forth.gr) [33] constitutes a simulation space of a real classroom, featuring both commercial and custom-made artifacts enhanced with state-of-the-art technologies aiming to support both students and educators throughout the entire educational process. In more detail, the classroom is equipped with a wall-to-wall projected interactive board, technologically-augmented student desks [34], a comfortable workstation for the educator, and various ambient facilities (e.g. motorized blinds, smart lights) that can be manually or automatically controlled to adjust the classroom atmosphere according to the learning context. Additionally, the students have access to programmable robots and X-Reality gadgets, through which they can explore different skills and practices [35].

One of the key classroom artefacts is the board. Research has shown that interactive boards can improve teaching and learning by providing access to a wide range of computer-based applications, capturing and maintaining a simultaneous focus of attention for large learner groups, supporting collaboration and encouraging discussion [36], [37]. The **Intelligent Classroom Board** of ICS-FORTH spans across two classroom walls (i.e. the wall in front of the entire class and one side wall) transforming them into large interactive displays.

1.4 Objectives of the CognitOS Classboard

This work aims to equip the **Intelligent Classroom Board** of ICS-FORTH with a sophisticated framework, named **CognitOS Classboard**, which takes advantage of the intelligent facilities offered by the environment (e.g. user localization service) and of the amenities offered by the wall-to-wall board in order to enhance educational activities and provide educator- and student-oriented experiences.

From the educators' perspective, the CognitOS Classboard aims to support lecturing. In particular, the main goal is to provide a unified working environment for the large interactive board, offering a natural and effortless interaction paradigm, which will allow educators to orchestrate their teaching assets in an optimal way (e.g. use different workspaces to place different types of applications) and present rich material to the students. Additionally, it aims to support course preparation before class, by equipping educators with user-friendly tools that will allow them to initialize the contents of the board with the appropriate material in advance.

Despite the fact that the main focus of this thesis is to support educators, the CognitOS Classboard can be beneficial for students too. In more detail, it aims to equip educators with appropriate tools to create highly engaging and fascinating learning experiences by transforming the classroom into an immersive environment on demand. Additionally, taking advantage of the multi-touch functionality offered by the Intelligent Classroom Board, the proposed system will be able to support student collaboration and thus enhance the learning process.

Aiming to be as less invasive as possible in the classroom environment and tone in with the already standardized processes of lecturing and teaching, the goal of CognitOS Classboard is to support multimodal interaction and follow well-established interaction metaphors (e.g. writing on a digital board should be similar to writing on a plain whiteboard). Towards that direction, taking advantage of the classroom's ambient facilities, multimodal interaction is supported through: i) touch on the wall, ii) mid-air gestures, iii) voice commands, and iv) user position tracking (e.g. the educator can select an application to follow him/her while moving around). Finally, the system is able to identify the height of the user and adjust the position of certain elements, so as to bring them within reach, which is also useful for students, especially children.

In more detail, through the "CognitOS Classboard" educators will be able to:

- Use an interactive marker to sketch, handwrite notes on the wall (as one would do on a common whiteboard), and interact with any of the UI elements.
- Organize the applications required for a specific lecture into a collection of workspaces (e.g. the Chemistry workspace contains the periodic table pinned at a certain spot on the wall).
- Create immersive experiences by selecting multimedia to be projected on the walls. Such a facility could enable interaction with the real world in ways that were not possible before (e.g. immerse students into a rainforest).
- Use X-Reality technologies to create highly compelling learning experiences (e.g. project the rivers of a country over its physical map), through which students can explore and improve different skills.
- Access useful educational applications (e.g. Quiz, Dictionary, Multimedia Viewer).

To summarize, the CognitOS Classroom: (i) offers a sophisticated working environment (i.e. workspaces), (ii) provides a tablet and a desktop application to configure workspaces a-priori, (iii) supports multimodal interaction, and (iv) permits the creation of immersive experiences.

1.5 Thesis Outline

The rest of this thesis is organized in the following way:

- Chapter 2 reviews related work regarding frameworks targeting Interactive Boards.
- Chapter 3 describes the design methodology that was followed, introduces several scenarios that motivated this work and outlines the functional and non-functional requirements of the proposed framework.
- Chapter 4 provides a description of the Intelligent Classroom that hosts the CognitOS Classboard.

- Chapter 5 describes the functionality of the system and the interaction paradigm.
- Chapter 6 elaborates on the implementation details.
- Chapter 7 describes the evaluation process and its results.
- Chapter 8 concludes the thesis with a summary of the results and a discussion of possible future directions.

Chapter 2

Related Work

This chapter reviews frameworks targeting Interactive Boards and afterwards presents the features that those frameworks support.

2.1 Hardware-specific solutions

EPSON has created an Interactive Projector to “transform teaching” [38]. This projector can display content on scalable screens up to 100” in Full HD. The classroom can become a dynamic learning environment (Figure 2) by supporting collaboration and virtual learning, while being able to project and share content between up to four different devices simultaneously. On the display, educators can highlight and annotate content to create an enhanced learning experience.

Figure 2: Education room illustration enhanced with EPSON technology [38].

ViewBoard [39] is an interactive whiteboard application available for large displays offering tools to enhance the learning process and engage students (Figure 3). These tools include general purpose utilities, such as a highlighter, an annotator and a virtual pen, as well as course-specific tools (e.g. rulers, a protractor and a caliper for the Mathematics course). The educator can also drop files in the application from the web or any other device using cloud storage systems. ViewBoard comes with two independent pens used for writing, annotating and highlighting. The two pens can be used at the same time, thus enabling collaboration between students. ViewBoard also features two sophisticated mechanisms; the first has the ability to recognize handwriting and transforms it into computer generated text and the second one recognizes free-hand drawings and searches for similar content in google images. Finally, educators can record their lectures and save the content to any cloud service, as well as distribute it to students' devices.

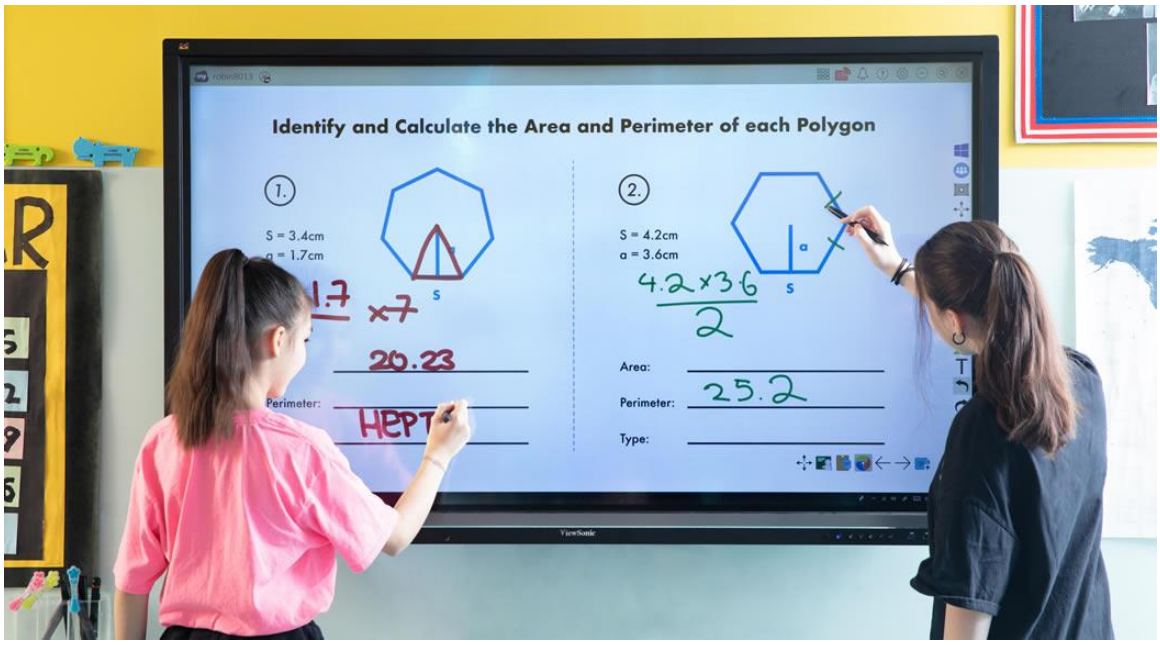


Figure 3: ViewBoard doubles the learning with two independent pens [39].

Figure 4: EZWrite is an easy and fun annotation solution [40] .

EZWrite [40] is an annotation application, exclusively installed on the BenQ's Interactive Flat Panel (IFP), which transforms it into an interactive whiteboard. By using the two different accompanying pens, students can collaborate either together (Figure 4) or with the educator, making the lecture a more fun experience. Additionally, a range of functions is available through a floating menu, including recording, screen capture, pen, eraser and whiteboard. EZWrite also features an Intelligent Handwriting Recognition mechanism, which can transform letters, numbers and shapes into legible materials. Also, the BenQ IFP comes with a video recording feature that allows educators to record lectures for class preparation or review purposes.

Furthermore, the content created during a lecture (e.g. annotation, notes, etc.) can be saved either as .pdf or .png files, and the educators can share them via emails, QR codes, USB drives or store them in the internal memory of the IFP. EZWrite offers a toolbox of useful utilities, which include a calculator and a geometry assistant (both of which identify free-hand writing and drawing and transform it into legible content), a timer and a collection of game-based features. These features include a collaboration space for up to three teams, a random selection mechanism for identifying which student should answer at

any point, a buzzer which hosts game show style quizzes and a score board to keep track of the teams' progress.

MagicIWB [41] is an all-in-one solution for an Interactive White Board without the need for an external PC, which is depicted in Figure 5. MagicIWB offers tools to write, draw, highlight and annotate content, which can be saved internally and shared across multiple devices for the students to view. MagicIWB supports multitouch interaction from up to ten different users, thus making collaboration and active learning a fun process.

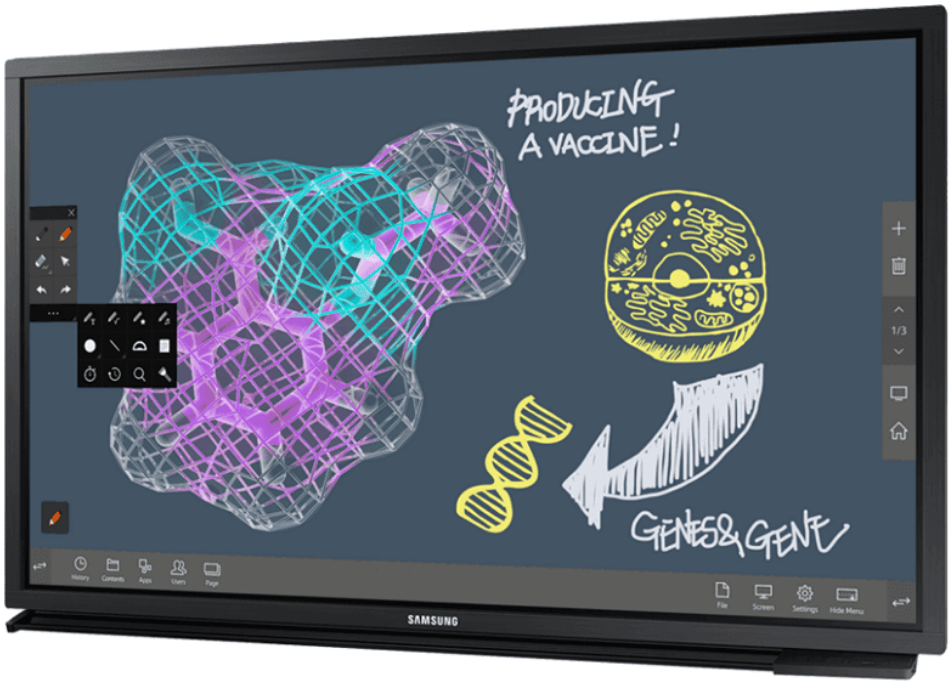


Figure 5: MagicIWB, Samsung's powerful, intuitive Interactive Whiteboard software [41].

Oktopus [42] is an interactive software designed for being used together with AVer CP Series IFPs (Interactive Flat Panels) [43]. It is able to simultaneously show contents on students' devices (Figure 6) to let them involve more in learning, while it also permits users to annotate over any application, take notes, answer questions, and share educational material.

Figure 6: Oktopus software supporting collaboration to enhance learning [42].

2.2 Hardware-independent solutions

The work in [44] introduces a **Classroom Window Manager** mechanism, a decision mechanism for adjusting application placement on various classroom artifacts, i.e. AmIDesk, SmartDesk, AmIBoard, SmartBoard (Figure 7). This mechanism provides a set of rules for designing applications in a specific manner to enable window placement on the various devices. In more detail, the SmartBoard and AmIBoard window managers permit designers to create applications that follow specific layouts (four in particular), so as to ensure reachability of interactive components. A set of tools is available, including an Application Switcher (a functionality similar to windows' task switcher) and an Application Launcher responsible for displaying shortcuts to course-specific applications. Finally, given that board artifacts might have more screen real estate available than student's personal devices, designers can exploit the "previews" feature to display snapshots of other information available through the same application.

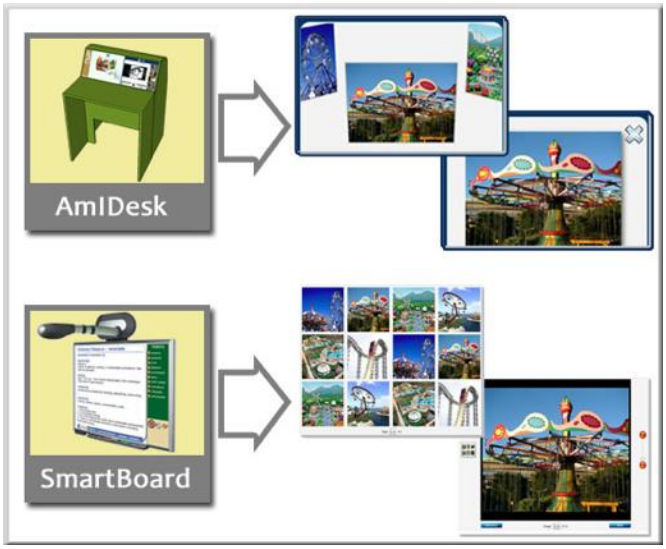


Figure 7; Adaptation of a multimedia application to meet the needs of different artifacts [44].

The work in [45] introduces **Intellichalk**, a multi-display digital blackboard system. The system uses four digital displays placed in a linear manner (Figure 8), creating an "infinity" canvas, which can be used to write and/or place images. A tablet and a stylus are used as input devices offering a set of different pen colors to write and annotate content. Images can be selected either from the internal storage of the tablet, an external storage device or the web. In addition, the system offers a collection of plugins, such as a gallery of graphic elements and handwriting recognition. The writing area on the input device extends vertically, to match human writing habits, however the content is translated horizontally on the displays, where the rightmost one serves as the focus area. The first three displays act as "history" areas, hosting older content, as the educator produces further content.

The system has been evaluated in the long-term during three different courses, i.e. two different Mathematics courses and a Computer Science course, and both educators' and students' perceptions were noted. The results of the evaluation indicated a positive attitude towards using the system in everyday lectures, since it was easier to follow the educator's thoughts as opposed to slide presentations.

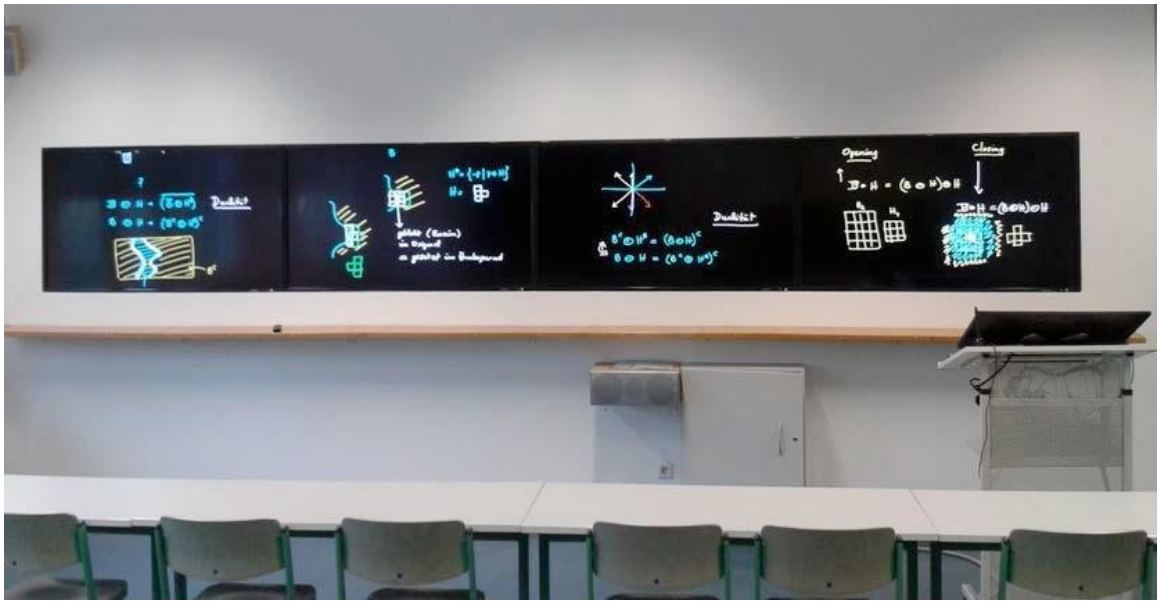


Figure 8: Intellichalk deployed in an intelligent classroom [46].

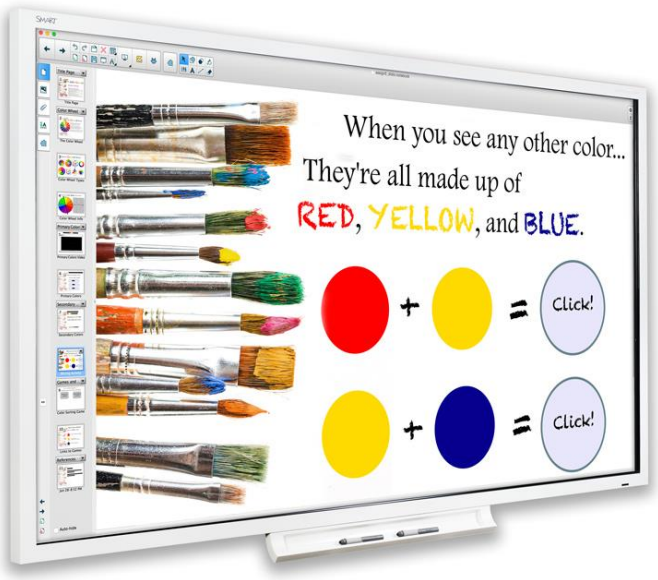


Figure 9: Smart learning suite software [47].

Smart Learning Suite [48] can transform static content into interactive experiences, providing tools for lesson creation, student assessment, collaborative workspaces, game-based learning tools and subject-specific features. The Suite is a desktop software, compatible with Windows and iOS (Figure 9). In addition to the desktop application, the Suite provides an online environment where educators can add interactive game-based features to a

variety of file types (e.g. PowerPoint, PDF) and send the lessons to students. Students can then either work individually or collaborate online to solve the exercises.

Figure 10: ActivInspire on the Promethean ActivPanel [49].

MimioStudio [50] is an educational software developed to enable educators to create interactive courses and keep track of student progress. MimioStudio is the software behind the MimioClassroom suite, which offers a collection of tools, such as a document camera and a pen tablet. MimioStudio can be also used with any interactive large display, aiming to replace the traditional classroom whiteboard with an interactive one. Through this software educator can create a variety of small-scale assessments for students, such as short-answer and short-essay questions. The students' answers are stored, graded and available for viewing by the educator in the featured MimioStudio Gradebook. Finally, MimioStudio enables collaboration by connecting up to three mobile devices – with the MimioStudio mobile up downloaded – and sharing student's work in front of the classroom.

ActivInspire [49] is a collaborative lesson delivery software for interactive displays (Figure 10). It offers a dashboard where educators can create

engaging lesson with compelling content using the application's vast suite of lesson creation tools. These lessons can be enhanced with external content, such as PowerPoint slides and PDF documents, and can embed multimedia files imported from existing resources. ActivInspire also supports multitouch interaction to encourage collaboration and enhance student engagement with features like interactive math tools. Educators can use ActivInspire to create interactive activities selecting from the available customizable templates for Matching, Flashcards, Crosswords and Memory games. These activities can be tailored to fit the needs of each respective course.

Figure 11: Canvus collaboration software equipped in an intelligent classroom [51].

Canvus [51] collaboration software from MultiTaction is a giant workspace (Figure 11) that is easily scalable to fit any display (e.g. laptops, tablets, smartphones, large displays, wall displays). Canvus enables multiple users to share and manipulate data at the same time with very short to non-existent latency. Users can participate in meetings or brainstorming sessions either remotely or on-site. Users can place various assets or handwritten or post-it notes on the workspace using a virtual keyboard, an IR pen for handwriting recognition or drag and drop any other content from the canvas. Additionally, each canvas workspace can be split into multiple workspaces, essentially by creating duplicates of the first workspaces, which enables smaller teams to work independently on Canvus. Each workspace can be displayed on different devices. Finally, Canvus can be integrated with any videoconferencing

platform, such as Zoom or Webex to engage agile team members with videoconferencing.

Lü Interactive Playground [52] features one or two projectors and 3D cameras in order to create engaging activities that make children move, play and learn. The 3D camera system turns the giant wall projections into touchscreens that can detect multiple objects. For example, Figure 12 presents a math game that requires students to use balls in order to hit the targets on the wall that correspond to the correct answer.

Figure 12: Interactive playground.

The work in [53] presents the **Virtual MultiBoard** (VMB), a combination between a traditional blackboard and slide presentations. VMB uses a large display as the main device, an interactive display as the input/annotation device and a smaller device (e.g. laptop) for the educator's view. In the large display three PowerPoint slides (or similar media, such as PDF files) are presented in a linear manner, from left to right. The left-most represents the current slide, where the focus of the educator resides at each moment, thus its dimensions are bigger than the other two slides. The next two slides represent the "history" or the two previous slides from the current, in order to enable all students to keep track of the educator – even those who require more time to keep notes. The second screen – the interactive display – is used by the educator to annotate and highlight the slides. These annotations are also displayed in real time on the main display and are pinned on the slides. Every

student with a laptop and network access can submit questions to the educator. The number of submitted questions – not the questions themselves – is displayed on the main screen to give feedback to the students that their questions have been submitted. On his/her personal device – the third screen – the educator can view those questions and decide to display them publicly to initiate discussion.

Finally, the VMB has the ability to record the lectures, or parts thereof, and either store them locally or send/broadcast them to the students via network.

Figure 13. OneScreen Annotate interactive whiteboard software [54].

OneScreen Annotate [54] is an Interactive White Board software featuring a variety of tools to annotate and enhance brainstorming sessions – or activities in general where multiple people are collaborating (Figure 13). Annotate can support up to 50 remote users in a shared whiteboard space. It offers unlimited content annotation capabilities using intuitive tools, such as pens, erasers, search engine assignment and customizable toolbars. Additionally, the workspace size is unlimited, which facilitates collaboration between remote

and on-site (or a combination of both) users. Through the drag and drop functionality, users can import multimedia files, either from the web or a cloud storage system. Also, work sessions can be saved, recorded and reloaded on demand. By recording a work session, the system keeps track of both image, audio and annotation simultaneously. Finally, the system is equipped with text, shape and handwriting recognition to transform even vague shapes into computer generated graphics.

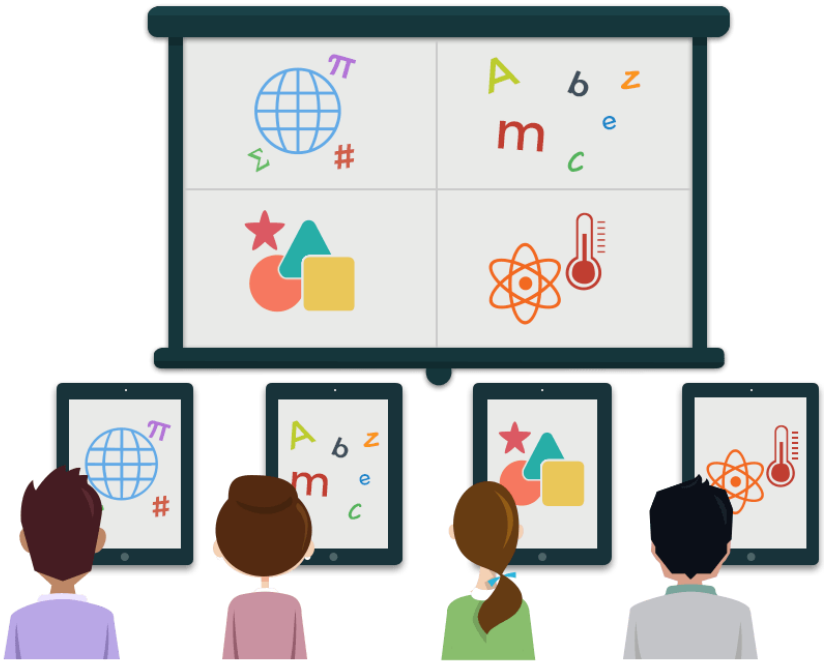


Figure 14: Annotate engage your entire class [55].

Annotate [55] is a cloud-based web app that transforms handheld or mobile devices into a mobile interactive whiteboard, by mirroring the device's screen on a projector (Figure 14). The educator interacts with his device as he would with an interactive whiteboard. The Annotate client offers a variety of drawing tools, and supports multitouch gestures to facilitate the educational process and make the course content dynamic. The educator can create assessments on-the-fly, while students can participate using their personal network-enabled mobile devices. Annotate automatically tabulates and grades student responses in real time, thus the educator can review student performance at glance without using precious lecture time.

2.3 Discussion

The literature review revealed several features that are supported by frameworks targeting Interactive Boards. Namely:

A. **Annotation**: writing notes or sketching on top of an application.
B. **Capture**: grab an instance of a specific board area.
C. **Remote manipulation** of board contents (e.g. via smartphone)
D. **Gestures** for quick control
E. **Collaboration support** via multitouch
F. **Multiple applications** open and visible at once
G. **Sharing Content** with students' devices
H. **Multiple Desktops** (or workspaces)
I. **Adaptation Facilities** (e.g. cater for various student heights)
J. **Easy window management** (e.g. follow me, summon)

Table 1 presents the features that each of the studied frameworks support. As seen, there is no system able to cover all the above features. The CognitOS Classboard aims to equip educators with appropriate tools (as described in A-J) that enable them to interact with the board in a natural and intuitive manner. It is worth noting, that it also introduces a number of innovative features (i.e. summon, follow-me, board manager, etc.) promoting the interaction with such a large display, that none of the existing systems have. Furthermore, the most important aspect that differentiates the CognitOS Classboard from the presented systems is its ability to communicate and collaborate with an Intelligent Classroom environment, in order to improve the overall User Experience and offer opportunities that isolated systems cannot provide. In more detail, it is able to consume data produced from the Classroom Environment, while it provides a public API that can be consumed by other services.

Table 1: Comparison Matrix of Board Software

	A	B	C	D	E	F	G	H	I	J
EPSON IP **[38]**	√				√	√				
ViewBoard **[39]**	√			√	√		√			
EZWrite **[40]**	√	√			√		√			
MagicIWB **[41]**	√	√			√		√			
Oktopus **[42]**	√						√			
PUPIL **[44]**				√	√		√		√	
Intellichalk **[45]**	√		√			√				
Smart Learning Suite **[48]**	√				√		√			
MimioStudio **[50]**							√			
ActivInspire **[56]**	√			√	√					
Canvus **[51]**	√		√		√	√		√		
Lü Interactive Playground **[52]**					√					
Virtual MultiBoard **[53]**	√		√				√			
OneScreen Annotate **[54]**	√	√			√	√				
Annotate **[55]**	√		√	√						
CognitOS Classboard	√	√	√	√	√	√	√	√	√	√

Chapter 3

Design Process

This Chapter initially introduces the methodology process, presents several scenarios that motivated this work and afterwards outlines the functional and non-functional requirements of the system.

3.1 Methodology

The process followed while designing the system, was the Design Thinking methodology [57]. In a series of meetings with several potential end-users (male and female users aged between 20 and 45 who have teaching experience), scenarios and personas were created for the 'Empathize' and 'Define' steps of Design Thinking (Figure 15). Next, for the 'Ideate' step, multiple brainstorming sessions were organized, and produced tens of ideas, which were then filtered through interviews with domain experts (i.e. software engineers, technical installation experts). Such process resulted in the identification and exclusion of the (currently) unfeasible ideas; for example, a lot of the ideas required the educator to perform hand-gestures, which are not supported by the Intelligent Classroom setup in its current form. Experienced interaction designers also reviewed the ideas and offered valuable insight and comments, as well as preferences in regards to which ideas had the most potential in their opinion

in terms of innovation, research interest and higher possibilities to be accepted by end-users (e.g. ideas that sounded attractive/cool/fun).

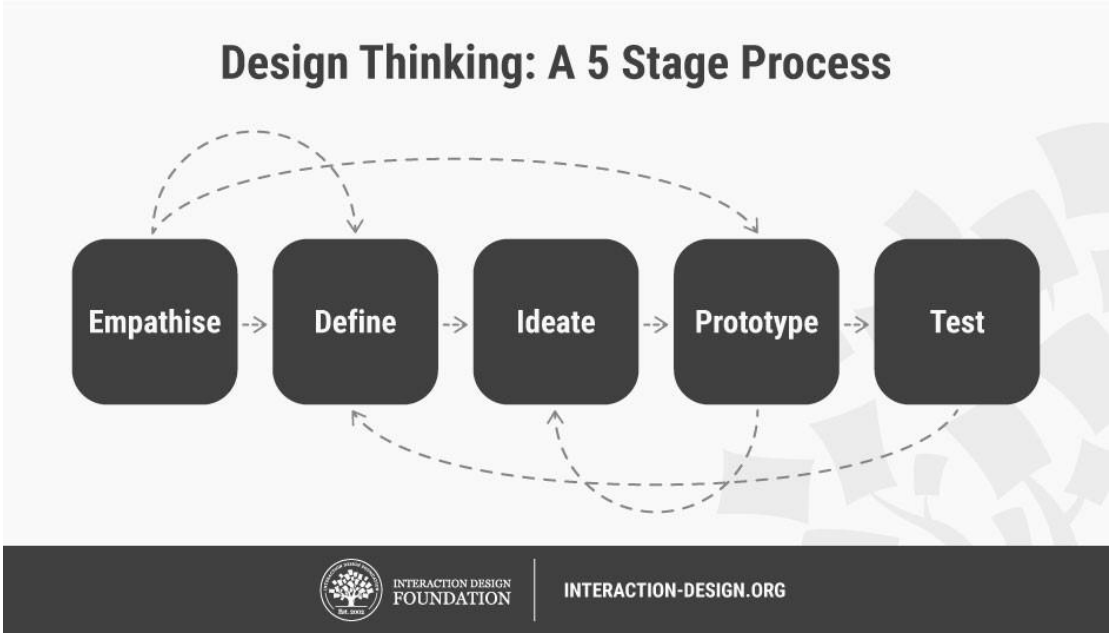


Figure 15: Design Thinking Process [58].

By properly shaping those ideas, a set of preliminary system requirements was extracted and constructed the final requirements of the CognitOS Classboard. In the Prototype phase, the design team followed an iterative design process where low (Figure 16) and high-fidelity (Figure 17) interactive prototypes were created for the most promising ideas (as resulted from the ideation phase). Moreover, before proceeding with user testing, a cognitive walkthrough evaluation experiment of the CognitOS Classboard was conducted with the participation of five (5) User Experience (UX) experts. The goal was to assess the overall concept and its potential, identify any unsupported features and uncover potential usability errors by noting the comments and general opinion of experts. Finally, after improving the system based of the findings of the cognitive walkthrough, a user-based evaluation was organized in order to observe real users interacting with CognitOS Classboard and gain valuable insights.

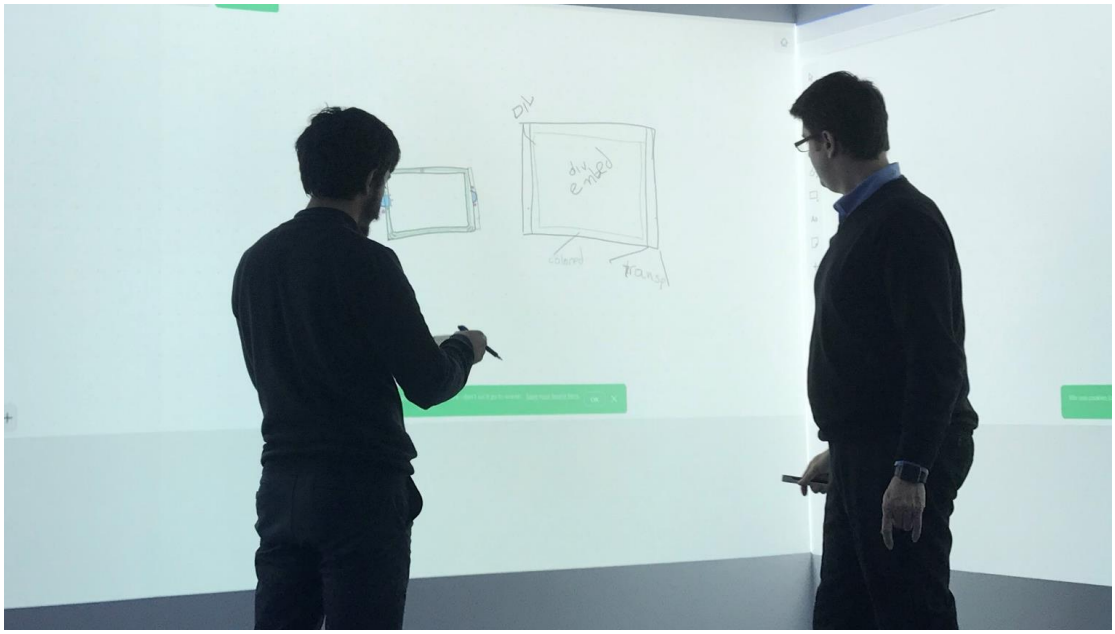


Figure 16: Designing lo-fidelity prototypes on the Interactive Classroom Board.

Figure 17: Hi-fidelity prototypes of the CognitOS Classboard.

3.2 Motivating Scenarios

Scenario building is a widely-used requirements elicitation method [59] that can systematically contribute to the process of developing requirements. Scenarios are characterizations of users and their tasks in a specified context, which offer concrete representations of a user working with a computer

system in order to achieve a particular goal. Their primary objective in the early phases of design is to generate end user requirements and usability aims. The following sections present a collection of envisioned scenarios where a University professor and a high school History teacher use the CognitOS Classboard during their lectures.

3.2.1 Teaching a University Class

Raymond teaches Human Computer Interaction at the Computer Science Department of the University of Crete. The department features an Intelligent Classroom that has been assigned to Raymond's course.

For today's lecture, Raymond wants to introduce the Process of Design Thinking. He has retrieved a couple of images and an interesting video that he wishes to share with his students. So, -in order to save time- before the lecture he uses the PC at his university office to prepare the contents of the board. To do so, he launches the "CognitOS Classboard Desktop" app and simply drags the multimedia and presentation files on the empty "Board". In order to present the video in full screen, without rearranging the remaining items on the Board, he decides to create a new "Board" and moves the respective file there. At 14:50 he leaves his office and moves towards the Intelligent Classroom to start his lecture.

He enters the classroom and greets a group of students who have already settled in. As he waits for the remaining students to join, Raymond launches the application that presents the course's schedule in order to remind the student's that they have an upcoming deadline.

Since this application presents secondary information, he uses the Workspace Manager tool in order to send it to the side wall of the classroom.

At this point all students have already arrived in the classroom, so Raymond greets them, reminds them of their assignment deadline and starts lecturing based on the PowerPoint presentation that he has prepared.

As he reaches slide number five that displays a representation of the Design Thinking Process, he captures an instance of the diagram and moves it on the

right side of the board. He then pins it at this location and moves the two pictures that he has previously loaded on the board near it. The first picture is related to the empathize step of the process (depicted on the image previously captured from the presentation file), so he uses the annotation functionality to make a connection between the two images.

He verbally explains the importance of that step and then "swipes up" to reveal the video player that he has already prepared to launch a film showcasing an experiment carried out in a few restaurants in São Paulo surprising customers with uncomfortable objects.

As soon as the video ends, he uses the same gesture to return to the "Primary Board" that contains the presentation. He then continues explaining the steps of the design thinking process. For the prototyping step, he uses the annotation functionality to make a connection between the step and the second image that is already on the board. Aiming to showcase the process of prototyping, he creates a new empty Board via the Workspace Manager tool, launches the application Whiteboard and sketches a login box. Next, he returns to the "Primary Board", captures the three images along with the annotation and sends the file to the students. Finally, Raymond moves towards the side wall and reveals the workspace containing the exercise submission application.

3.2.2 Teaching a History Class

Oliver teaches History at the 4th Heraklion High School. The school features an Intelligent Classroom that has been assigned to Oliver's course.

For today's lecture, Oliver wants to introduce the Battle of Thermopylae. He has retrieved a couple of interesting images and a period-specific map that he wishes to share with his students. Oliver has also created a PowerPoint presentation to assist his lecture. So, -in order to save time- before the lecture he uses the PC at his school office to prepare the contents of the board. To do so, he launches the "CognitOS Classboard Desktop" app, creates a new workspace and simply drags the multimedia and presentation files on the

empty workspace. At 14:50 he leaves his office and moves towards the Intelligent Classroom to start his lecture.

He enters the classroom and greets his students; he then opens the Board menu and selects to launch the Presentation Viewer app in order to begin lecturing on the presentation he has prepared. Additionally, from the same menu, he launches the Multimedia Viewer and initiates a slideshow of notable pictures and illustrations that would enhance his lecture.

As he reaches slide number twenty-five that displays distinctive information about the battle of Thermopylae, he uses the Board's Annotation tool to highlight important points he wishes to emphasize. At that point he launches the Map application from the Board's menu and zooms on the location of the battle. The Board uses the virtual map to display a video projection of the advances of opposing forces on the Greek army.

As the animation on the map reaches the end, Oliver decides to initiate an impromptu verbal mini-quiz, so he launches the Quiz application and asks his students to identify the most infamous quote of the battle of Thermopylae. When the correct answer is given, he proceeds to write it down on the Whiteboard, using his virtual marker.

Finally, as the time reaches 16:00, the clock on the Board rings, to signify the end of the lecture. Before greeting the students, Oliver draws their attention to the side wall and informs them about the topic of their next lecture, as displayed on the Calendar application. As the lecture comes to an end, Oliver captures a screenshot of the Board's contents and shares them along with all course's material to his students' devices through the Board's sharing service.

3.3.3 Meeting Room Presentation

The technology company Mobile-x which is located in Athens wants to introduce a new updated version of their mobile device. Today, the executives of the company have organized a meeting with the sales and marketing managers regarding the sale and advertising campaign of the product.

For today's meeting both Niki the sales manager and Ryan the marketing manager prepare the content of their presentations using the Desktop application.

The meeting is about to start, therefore Niki makes a gesture "S" to launch the board and loads the prepared content. On the main wall the presentation file, which concerns information about the past sales of the previous mobile model, is shown. On the other hand, the side wall displays corresponding graphs and plots, as well as the key points to be discussed in the meeting. As she reaches slide twenty containing a detailed table regarding last year's monthly sales, she uses the annotation tool to highlight the months with the lowest revenue. Niki then summons the plot containing the monthly expenses on advertising and associates it with the decreased sales, concluding that when the monthly expenditure on advertising is low revenue shrinks. To refer to this information later, she takes a screenshot of the content.

Niki continues with the presentation of the company's sales data, on the slide twenty-five she lists statistics related to the sellings per each store in the area of Athens. To illustrate these findings, she has already launched the map application on the second board and has added information correlating these statistics with each store location. To switch between boards, she uses the paging component on the top area of the board. During the presentation, one of the participants asks her the total amount of the purchased devices in the area of Athens, at which point she launches the calculator application and computes the answer. The first part of the presentation is completed and the participants take a five minutes break.

In the second part of the meeting, Ryan proceeds with the presentation of the advertising campaign of the product. Ryan is shorter than Niki, and he is left-handed so the content of the board is adjusted automatically to his physical characteristics (e.g. the windows' menu is adjusted on the left side, the windows are placed on a lower point in the board). He carries on with presenting general information about the campaign. In the next step, he explains the positive impact of advertising their product on monthly sales, and

then “swipes up” to reveal the video player to introduce the three proposed videos and exhibit the five prospective posters (using the image viewer) for the campaign. Concluding his presentation, he asks the participants to vote and choose one of the proposed videos and one of the proposed posters accordingly. They have five minutes to vote, so Ryan launches the countdown timer application. The voting session is completed, Ryan moves on the side wall and reveals the results of the voting on the alternative view of the workspace.

After the completion of both presentations, Niki also stands up next to Ryan in order for the Q&A session to begin. Some executives inquire for a few clarifications and each question is shown as a bubble adjacent to the physical position of the presenters. The Q&A session is concluded, Niki then displays the deadline’s timeline on the side wall, opens the calendar application to inform the executives of the next scheduled meeting and shares the event on their calendar.

The meeting is now completed, so Niki makes a “C” gesture to switch off the board.

3.3 Requirements

This section presents the high-level functional and non-functional requirements that CognitOS Classboard and the related front-end helping tools (i.e. desktop, tablet, mobile applications) need to satisfy. Such requirements have been collected through an extensive literature review and an iterative elicitation process based on multiple collection methods, including: brainstorming, focus groups, observation and scenario building.

3.3.1 Functional Requirements

FR-1: Educators should be able to launch educational applications.

FR-2: Educators should be provided a mechanism to organize the applications in smaller collections, so as to manipulate them as a group.

FR-3: Educators should be able to alternate amongst several boards (similar to the multiple desktops of Windows 10) so as to enable a more versatile working area.

FR-4: Educators should have access to primary course material (e.g. presentation files), as well as secondary applications (e.g. maps, clock, weather).

FR-5: Users (educators or students) should be able to open, close, maximize, pin/unpin, resize and move application windows in a quick and easy way.

FR-6: Before the beginning of the lesson, the educator should be able to configure the board's state and pre-load the material of the course (e.g. set up the initial state of the board by launching applications with specific content and arranging them as needed).

FR-7: Users (educators or students) should be able to access application windows positioned away from them, even if they stay relatively close to the board.

FR-8: Educators or students should be able to pin an application window at a specific location on the board so as to ensure that it will hold its position regardless of other windows rearrangements that may occur.

FR-9: The educator should be able to set an application to follow him along the board, so that he has direct access to it.

FR-10: Educators or students should be able to call an application window to relocate to a board area closer to their physical location in the classroom.

FR-11: The educator should be able to personalize the board contents according to their preferences and the current course's needs during the lecture (e.g. launch auxiliary applications, load specific content depending on the lesson's material).

FR-12: Educators should have the opportunity to keep track of every application launched on the board, even if they are standing relatively close, being away from the board or sitting at their desk.

FR-13: Educators should be able to interact with the system using multimodal interactions (e.g. mid-air gestures, voice commands).

FR-14: Users (educators and students) should be able to interact with the board either with their hands (i.e. touch) or with the use of interactive markers.

FR-15: Users (educators and students) should be able to handwrite notes on the board (as one would do on a common whiteboard).

FR-16: The system should ensure that interactivity is not affected due to the physical characteristics of the users (e.g. younger students are usually shorter than their educators).

FR-17: The system should identify the type (i.e. educator or student) of each user, so as to personalize the content and functionality to deliver.

FR-18: Multiple users should be able to interact simultaneously with the content of the board, collaborating on the same or different applications.

FR-19: In the case that multiple users interact with the board at the same time, the system should indicate what each user is in control of.

FR-20: Educators should be able to choose at any given time whether students have access to interact with the board.

FR-21: The system should attract students' attention to a specific area on the board (e.g. towards a specific application).

FR-22: Educators should be able to annotate the content of the board (e.g. circle an important point during on a presentation file, annotate on top of two or more application windows).

FR-23: Educators or students should be able to capture a snapshot of the entire board, the contents of a selected application or a specific area on the board.

FR-24: Educators should be able to share content with students (e.g. capture a snapshot and forward it).

FR-25: Educators should be able to perform simple gestures (e.g. swipe left/right to change the slide during a presentation), in order to facilitate interaction with the board.

FR-26: Users should not get distracted by the vast display area during a lecture. To this end, unfocused projected areas should remain dark.

FR-27: Educators should be able to select appropriate multimedia in order to create immersive experiences (e.g. immerse students into a rainforest), that is why an application should be able to maximize and cover all the board.

FR-28: Software engineers should be able to develop new applications and import them in the system.

3.3.2 Non-Functional Requirements

NFR-1: Acceptance Testing Requirements

A full-scale user-based evaluation should be carried out to ensure educators' acceptance. Participants should have varying levels of experience using web or mobile applications. The overall score of a Standard Usability Scale (SUS) based post-evaluation questionnaire should be above 75%.

NFR-2: Documentation Requirements

A Quick Start Guide should be provided.

NFR-3: Platform Compatibility Requirements

From a user perspective, the front-end tools (i.e. PC, Tablet, Mobile applications) should be accessible via web browser.

NFR-4: Maintainability Requirements

CognitOS Classboard should permit easy maintenance in the sense that faulty or worn-out components should be repaired or replaced without having to replace still working parts and any updates should be verified and validated before their final deployment. All those updates should not affect the installed applications. Simultaneously, when an application is updated the rest system should not be affected.

NFR-5: Deployment Requirements

CognitOS Classboard should minimize any deployment requirements for its core components. During system's deployment the installed applications should not be affected, and vice versa. Additionally, it should offer the necessary facilities that will automate the deployment process for any application.

NFR-6: Interface Requirements

Software interface requirements include dealing with an existing software system, or any interface. In more detail, the CognitOS Classboard framework should respect and adhere to the formal specifications of any external provided applications. At the same time, the board should provide a mechanism to enable communication with the installed applications, in order to retrieve information about their state and their functionality, so as to appropriately use it (i.e. restore an application's state if needed, provide shortcuts to invoke remote functions).

Chapter 4

The Classroom behind CognitOS Classboard

This chapter describes the software and hardware infrastructure of the Intelligent Classroom of FORTH-ICS.

4.1 Software Infrastructure

The software architecture of the Intelligent Classroom follows a stack-based model where the first layer, namely the AmI-Solertis middleware infrastructure [60], is responsible for (i) the collection, analysis and storage of the metadata regarding the environment's artifacts and (ii) their deployment, execution and monitoring to create a ubiquitous ecosystem. The next two layers, namely the ClassMATE and the LECTOR frameworks, expose the core libraries and finally the remaining layer contains the end-user applications responsible for delivering educational interventions and accepting user input. As far as the end-user applications are concerned, CognitOS [61] delivers to the students a sophisticated environment for educational applications hosting able to present teachers' interventions.

4.1.1 AmI Solertis

Building services for Ambient Intelligence environments implies that multiple different technologies and protocols will be used by the various technological components in order to define and expose their functionality. The deciding factor on which specific protocol will be used, in addition to any prospective standards and guidelines that suggest certain approaches [62], [63], is their technical capabilities from a hardware (e.g. network interfaces, processing power, battery-based operation) and a software (e.g. OS, runtime environment) perspective.

AmI-Solertis [64] enables fast, easy and error-free integration of **external AmI artefacts** (i.e. services) independently of their type (i.e. back-end, front-end or mixed services that follow the Software-as-a-Service paradigm [65]). Moreover, it supports the creation of **AmI scripts** that define the behaviour of the technological facilities (i.e. business logic) towards creating pervasive, intelligent and personalized environment experiences by combining multiple components. The AmI-Solertis system is built using a micro-service architecture style that enables it to be used as a backbone [66] across a wide range of ubiquitous systems [67] and intelligent environments with diverse objectives (e.g. compose a new compound service using existing ones, define the behaviour of a smart hotel room [68], control a smart home, build an intelligent management system for a smart city [69]).

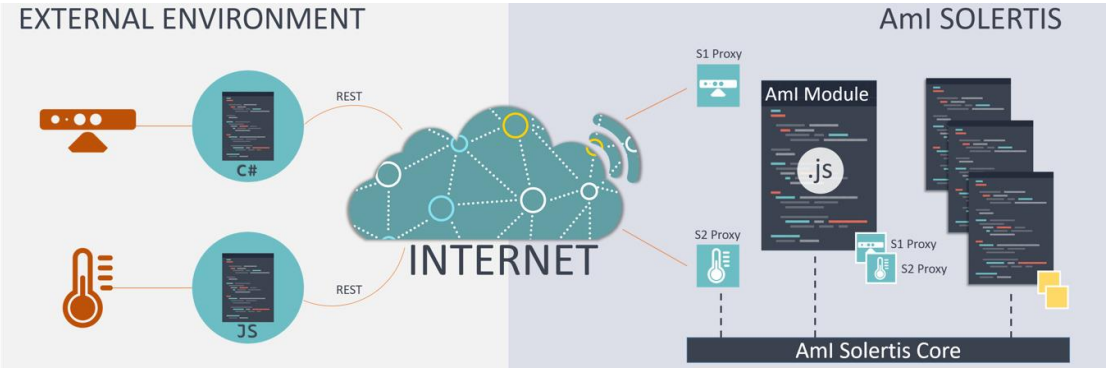


Figure 18: The AmI-Solertis Hybrid Communication Protocol [64] .

Building on the benefits of asynchronous and event-based communication [70], [71, p.], [72], AmI-Solertis introduces a unified Hybrid Communication

protocol that combines the widely-used Representational State Transfer (REST) [73] and the OpenAPI Specification (OAS) [74] with asynchronous and event-based communication facilities to integrate heterogenous services in a standardized -yet agnostic- manner (Figure 18). Therefore, an AmI artefact or an AmI script on the one hand exposes a REST interface to receive incoming calls, and on the other hand communicates its intention to the AmI ecosystem by emitting appropriate events via the AmI-Solertis Event Federator.

AmI-Solertis encapsulates the complexity of configuring and performing remote calls in automatically generated proxies that eliminate the difficulties of distributed programming by (i) masking remote operations into local methods, and (ii) enabling consumers to register their interest to events coming from a remote component without specifying any details about the underlying topology and infrastructure. In addition to code minimization, proxies also empower AmI-Solertis to dynamically adapt and adjust the invocation process in order to address emerging requirements such as re-routing a call to a replicated host to achieve load balancing, immediately terminating a call if the remote endpoint is unavailable, intercepting a call and logging relevant QoS-related metrics, replacing a target endpoint with another that offers semantically similar functionality. In addition to AmI components management (i.e. AmI artefacts or scripts), AmI-Solertis offers an online IDE, named AmI-Solertis Studio, which aims to assist developers in creating, exploring, deploying, and optimizing the AmI scripts (i.e. programs) that control the behaviour of the AmI environment by combining and orchestrating various AmI artefacts or other AmI scripts that reside in the ecosystem.

4.1.2 ClassMATE

The ClassMATE [75] framework is an integrated architecture for Ambient Intelligence environments that monitors the environment and makes context-aware decisions in order to assist the student in conducting learning activities, and the educator with administrative issues. ClassMATE features a

sophisticated, unobtrusive profiling mechanism that facilitates the classroom's students' behavior monitoring and assessment, in order to provide user related data to the classroom's services and applications. Finally, it encapsulates a sophisticated content discovery and personalized content delivery mechanism. In more detail, an Ontology-based Aggregator, namely the DataSpace, belongs to the core components of the ClassMATE framework, and aims to deliver personalized educational content to each individual learner based on his current needs and context of use. DataSpace, incorporates mechanisms to: (i) semantically classify and archive learning objects based on their metadata, and (ii) personalize content delivery based on each learner's needs and preferences.

4.1.3 LECTOR

LECTOR is a framework developed based on the sense-think-act cycle, a process which according to cognitive psychology stems from the natural human behaviour to receive input from the environment (perception), process the received information (thinking), and act upon the decision reached (behaviour) [76]. The main responsibilities of LECTOR are: (i) monitoring human behaviour inside an Intelligent Classroom, (ii) detecting problematic situations, and (iii) intervene appropriately to each situation. In more details, LECTOR – in conjunction with the AmI technologies of the Intelligent Classroom – observes student attention levels during lectures (SENSE) in order to identify possible inattentive behaviours (THINK), and provides situationally appropriate interventions to support both students and educators in the learning process (ACT).

Additionally, the AmI facilities enable LECTOR to continuously SENSE students' behaviors inside the classroom environment, a foundation that led to the extension of the SENSE-THINK-ACT model of the framework with the notion of LEARN. By being able to observe student behavior both before and after an intervention, LECTOR acquires useful feedback on the efficacy of each selected intervention. More specifically, the LEARN component (i) utilizes this

knowledge of efficacy to auto-rank the interventions accordingly, while also (ii) incorporating end-user feedback on the acceptance of each intervention, which enable the fine tuning of LECTOR's decision-making mechanism to provide more appropriate interventions.

LECTOR can be programmed using a three-step process of simple "if this, then that" rules to connect behaviors with interventions, following the trending trigger-action model [77]–[79], for programming AmI environments. There are five core concepts of the LECTOR's approach: (i) rules; models for binding a behavior with an intervention via a trigger, (ii) behaviors; models of user or device actions, (iii) triggers; models of high-level behaviors that can initiate an intervention, (iv) interventions; actions selected through the system's sophisticated decision-making mechanism that aim to support users and (v) intervention hosts; physical artifacts that either display applications with carefully curated content or control the physical environment.

The process begins with step one (1); definition of a behavior (e.g. a student yawns). Step two (2) defines the conditions under which a behavior becomes a trigger (e.g. the behavior is happening repeatedly during a lecture). Finally, step three (3) connects the triggering behavior to an appropriate intervention (e.g. notify the educator to alternate their pedagogies to regain student attention).

This three-step process offers great scalability and flexible rule management, mainly due to the fact that each necessary component (i.e. behavior, trigger and intervention) is defined in isolation and then connected only in terms of their combined outcome. This segmented approach – inspired by how an Application Programming Interface (API) simplifies programming exposing only the objects the developer needs – allows the independent modification of each element, facilitating collaboration between different users, since the connection points of the rules are always their outcomes.

Finally, in order to enable both developers and non-technical users to easily create such rules, a sophisticated, user-friendly authoring tool was developed, LECTORstudio [80]. LECTORstudio provides a simple an intuitive User

Interface (UI) to create rules through building blocks, which represent the various components of LECTOR (i.e. behavior, trigger and intervention).

4.1.4 CognitOS

CognitOS [61] is a student-centric working environment of adaptive educational applications for the Intelligent Classroom. CognitOS is permanently deployed in the in-vitro simulation of an Intelligent Classroom in the Human – Computer Laboratory of ICS – FORTH. The simulator features custom-made technologically augmented desks, each equipped with a 27-inch multitouch-enabled All-in-One PC, along with various sensors (e.g. eye trackers, cameras, microphones, etc.), the students' personal tablets and the educators' smartwatch or smartphone.

CognitOS educational applications can be utilized by LECTOR as means to present appropriate interventions, when deemed appropriate. For example, a mini quiz application can be launched explicitly by a student, when dictated by the educator, by selecting an appropriate exercise on her book, or automatically by LECTOR as a fun intervention to regain her attention during a reading assignment.

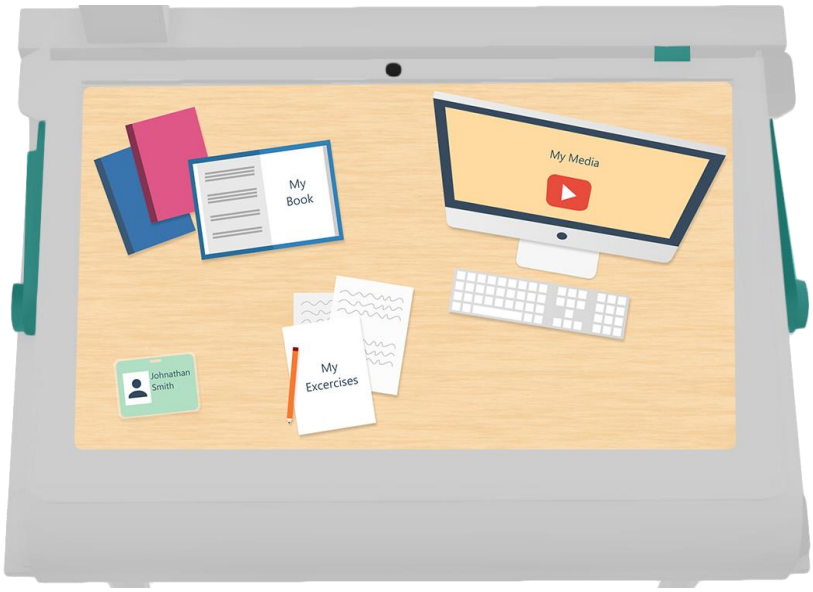


Figure 19: The desktop of CognitOS running on the augmented school desk.

The Desktop (Figure 19) is the main working area of CognitOS, which follows the metaphor of an actual desk containing various student items, such as

books, notes and pencils, that can be used to launch respective applications. More specifically, the Desktop contains:

(i) A pile of books, which offers a shortcut to the student's collection of books. Upon launching, the book related to the current course is displayed with the respective content.

(ii) A pile of notebook pages, where a collection of the student's exercises and homework is hosted, categorized per course and according to their status (i.e., pending, completed, graded). The exercises are filtered upon launching to display only course-related content.

(iii) An id card, which hosts the student's profile. Apart from several personal information (e.g., student name, age, grade) a complete gradebook of student's grades is provided, as well as a breakdown of the grades with further statistics on each course.

(iv) A computer monitor, which acts as a launcher to a multimedia player for presenting the respective content (i.e., videos, images and sounds).

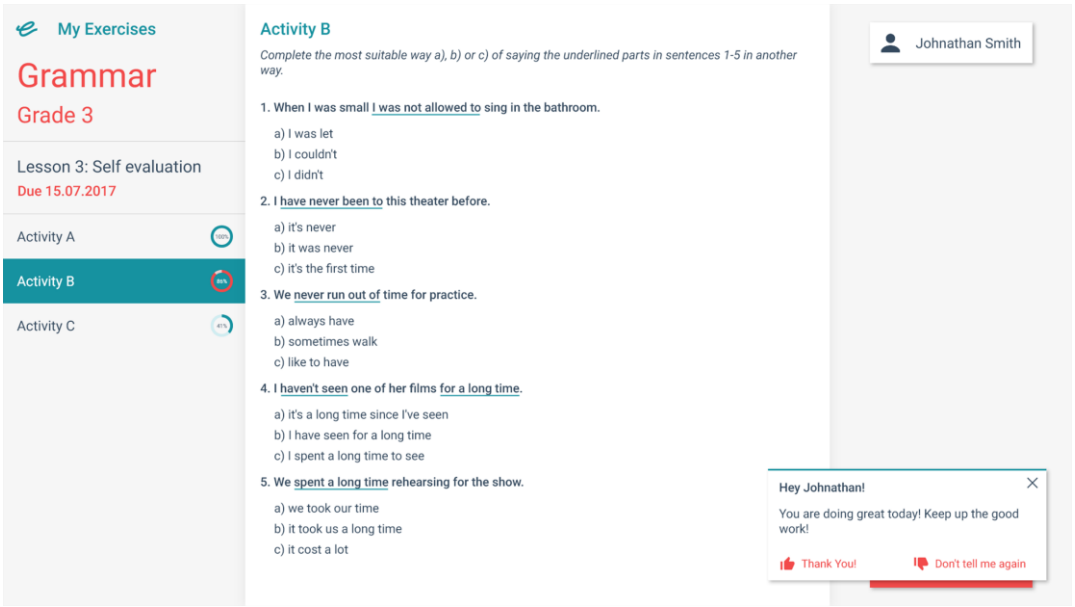


Figure 20: A notification appears trying to encourage a student during exercise solving

Inside an educational environment, students should be allowed to utilize more than one course-related applications simultaneously (e.g., exercises,

calculators, notes, etc.), hence a mechanism that allows multiple applications to be displayed, which also is able to identify and decide the best placement for each newly launched application is required. To this end, CognitOS is equipped with a sophisticated algorithm, which is responsible for window management and adjusting the applications according to several defined rules. These rules include (i) the placement of related applications; if an application is launched to provide additional material to an already existing one, they should be always displayed next to each other, (ii) the application with which the user is interacting or interacted last should remain on the foreground, (iii) secondary applications, such as calculator, calendar, etc. should be displayed on the student's tablet, when present, otherwise they should occupy less screen real-estate on the Desktop when more important applications are already launched. However, the window manager mechanism of CognitOS permits the manual rearrangement of the launched applications, so each student can customize and personalize their working environment.

Furthermore, in addition to its window management capabilities, CognitOS – in conjunction with LECTOR and AmI-Solertis – can deliver four types of interventions, namely notices, augmentations, alterations and restrictions. CognitOS features an advanced notification mechanism which can deliver personalized appropriate messages (i.e., notices) when LECTOR decides that a student seems unmotivated, troubled or disengaged from the current task (Figure 20). Additionally, each of CognitOS educational applications can be utilized to present appropriate interventions with appropriate content. These applications can either be launched on mini- or full-view, where the first is employed to present auxiliary content alongside the main application (i.e., augmentation) in order to support or motivate students, and the later aims to monopolize student attention (i.e., alteration) when a radical intervention is needed (e.g., students do not pay attention to the educator, are talking to each other, or are otherwise disengaged). Finally, each of CognitOS applications can be locked on demand to restrict access to content that is not relevant to

the current course or activity (i.e., restrictions), such as when a classroom-wide quiz is taking place.

4.2 Artefacts of the Intelligent Classroom

With the emergence of Ambient Intelligence, many challenges have surfaced concerning the equipment of the physical environment with technologically enhanced artefacts. When installed into an environment, such artefacts should reduce their overall footprint and blend harmoniously into their surroundings. In order to accomplish such unobtrusiveness, the artefacts are usually hidden or embedded in their surroundings and traditional furniture to become unnoticeable. The "Intelligent Classroom" (visualized in Figure 21), following these principles, features a collection of interdependent artefacts that are capable of exchanging information and communicating with each other offering a unified interaction experience. There are two categories of available artefacts, namely commercial equipment and technologically augmented everyday objects.

Figure 21: The Intelligent Classroom of FORTH-ICS [33].

Commercial equipment includes a variety of commercial **devices** (e.g. Philips Hue Lights [81], blinds [82], Alexa [83]) and **appliances** (Humidity, Ventilation, Air-Condition - HVAC), which are controlled either via their own Application

Programming Interface (API) or utilizing dedicated solutions (e.g. KNX bridge [84]).

Technologically augmented everyday objects include an Intelligent Student Desk, an Interactive Classroom Board, and an Educator's Workstation. Their characteristics are outlined in the following sections.

4.2.1 Intelligent Student Desk

The Intelligent Student Desk is one of the key classroom artefacts, where students spend most of their class time. Since its first conception in 2010 [85], when it featured a vision-based back-projected multi-touch screen, it has been remarkably re-designed and modernized. Today, the layout of the desk artefact has been upgraded featuring a more appealing and ergonomic design, embedding a 24-inch wide all-in-one computer, secured in a rotatable steel frame [34]. Currently, the next version of the student desk is under development. It will feature a modular design, where customizable surfaces could be added or removed on demand to support the specific and different needs of each course. For example, during the chemistry course the student can utilize a surface featuring equipment appropriate for chemical experiments such as smart flasks, test tubes, scales, a Bunsen burner and other basic accessories. This new generation desk aims to further enhance students' engagement and motivation, providing hands-on experience and offering personal study spaces with specialized equipment.

4.2.2 Interactive Classroom Board

Similarly, to the student's desk, the Intelligent Classroom Board has evolved through the years. The first version featured a commercially available interactive white board, while the second version used a 70-inch, 4K TV including a multi-touch panel with the aim of enabling collaboration and increasing the interactive area. The current version of the classroom board was inspired by the recent advancements in projection technologies, which can convert plain walls into interactive displays. To this end, extending beyond digital devices, two out of the four walls of the "Intelligent Classroom"

(i.e. the wall in front of the entire class and one side wall) act as interactive smart boards, where educational content (e.g. multimedia, notes, exercises) can be displayed. Moreover, the available projection area can be used to create appealing situational (i.e. course-, topic- and discussion-specific) environments capable of immersing students into real world situations (e.g. a museum, a historical event).

4.2.3 Educator's Workstation

The educator should be able to monitor and manipulate every aspect of the Intelligent Classroom (e.g. ambient facilities, educational software, intelligent behavior, automations). A comfortable workstation, namely an arm-chair with two embedded tablets and a variety of sensors is under development, aiming to assist educators during their mission of disseminating knowledge. The first tablet of the custom-made "intelligent" arm-chair is mounted on a rotating plastic arm, which the educator can manipulate so as to bring the device in front of them or hide it when necessary. This tablet features various applications that permit educators to have access to educational material, get a general overview of the class (i.e. classroom statistics, attendance, submitted assignments), and monitor student attention in a seamless and unobtrusive manner. Furthermore, another tablet is embedded on the left armrest acting as a secondary screen presenting auxiliary information, offering various utilities (e.g. dashboard with shortcuts to useful applications), and acting as a controller for the room's conditions (i.e. temperature, humidity, lights) and the various intelligent classroom artefacts (i.e. student's desks, board, walls, windows). At the same time, hidden LED lights permit educators to light-up the chair using color indicating various situations (e.g. lights of orange color indicate that students are expected to submit their assignments). Additionally, a force resistive sensor allows the environment to know whether someone is seated in the arm-chair, while embedded microphones support voice recognition, and directional speakers enable the delivery of sound notifications targeting the educator.

Chapter 5

The CognitOS Classboard

The CognitOS Classboard is an advanced window manager (i.e. "*a system software that controls the placement and the appearance of windows within a windowing system in a graphical user interface*" as defined by [86]), that facilitates both on-surface and above-surface interaction for large displays (e.g. wall-to-wall projections). In particular, it aims to transform the wall of an Intelligent Classroom into an interactive large board that can host educational applications and display appropriate multimedia content that can immerse students into real world situations. The proposed system: (i) offers a sophisticated workspace management mechanism, (ii) provides a desktop application to configure workspaces a-priori, (iii) provides a tablet and a mobile application to control the workspaces during the lecture, (iv) supports multimodal interaction, and (v) permits the creation of immersive experiences.

5.1 Design Challenges

During the design process of the CognitOS Classboard, an important issue that had to be addressed was to identify sophisticated mechanisms that permit educators and students to view and interact with the available applications in an easy manner, given the affordances of the large displayed area. To this end,

the design team encountered common challenges that often arise when designing for large public displays [87], such as:

Challenge A. Considering that it is difficult for users to access objects scattered around on a wall-sized display, especially when they tend to stay relatively close to it, what is the most efficient way for students and educators to close or move application windows positioned away from them?

Challenge B. What is the most efficient way for an educator to keep track of every application launched on the board, given that in some cases he/she might be standing relatively close to it?

Challenge C. What is the most efficient way for students and educators to open, close and move applications in a quick and easy manner? How can the system ensure that interactivity isn't affected due to the users' physical characteristics (e.g. teachers are usually taller than their younger students)?

Challenge D. (i) working up close to the display (e.g. performing tasks involving dealing with detailed information) and (ii) performing tasks from a distance (e.g. sorting photos and pages or presenting large drawings to a group) [87], what are the most appropriate techniques allowing a smooth transition from up-close interaction to interaction at a distance and vice versa?

Challenge E. Considering that both educators and students are expected to interact with the board, how can the system distinguish the type of each user so as to personalize the content and functionality to deliver?

Challenge F. Considering that multiple users might interact with the board at the same time, what is the most efficient way to clearly indicate what each user is in control of?

Challenge G. Given the size of the projected area (the wall in front of the entire class and one side wall), what is the most efficient way to draw the students' attention to a specific area on it, where the active application resides?

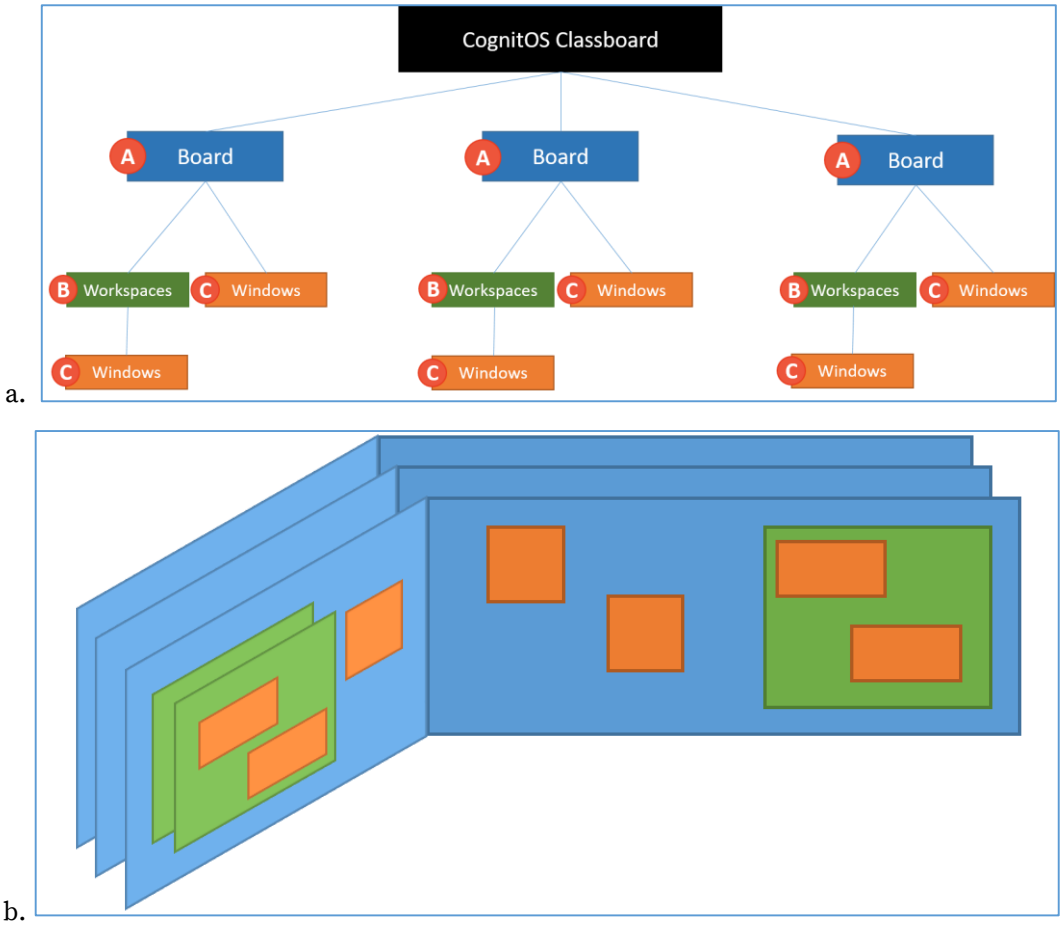


Figure 22: a. Representation of the CognitOS Classboard components hierarchy, and b. Visualization of a sample system instantiation

5.2 Fundamental Components

Given the aforementioned issues, the followed approach focused on creating a sophisticated board manager - appropriate for large displays - that supports the arrangement of educational applications during a lecture. Towards that direction, four concepts are introduced, namely (A) Board, (B) Workspace, (C) Window and (D) Applications (Figure 22).

A. Board. A Board refers to the entire surface that is used to project educational material. In more detail, the Board can be displayed on one or more physical classroom walls. It hosts Workspaces and Windows. Users can create unlimited number of Boards, while only one Board is visible at a time. That way, when educators alternate amongst the available Boards (which are similar to the multiple desktops of Windows 10) students are resented with entirely new content across the CognitOS Classboard.

B. **Workspace.** A Workspace is an area of a Board, which can host Windows. The existence of Workspaces facilitates educators to organize the application windows into groups and manipulate them concurrently (e.g. move them together).

C. **Window.** A Window is the graphical control element that hosts Applications. It can be moved, resized and pinned, while it offers various extra features that will be described later.

D. **Applications.** In order to support educators during a lecture and enhance the learning process, an application library is provided, consisting of built-in and external applications.

Figure 23: Representation of Alternative Workspaces.

5.3 CognitOS Classboard Functionality

5.3.1 CognitOS Classboard

The CognitOS Classboard can host educational applications (e.g. whiteboard, presentation viewer, video player, exercise viewer, etc.) -organized into different Boards and Workspaces- and offers a set of tools (e.g. menu, board manager, screen capturer, screen recorder, annotator). By using these tools, educators can easily manipulate (e.g. resize, close, hide) Boards, Workspaces and Windows, can create new Boards and Workspaces, launch new applications and arrange the position of existing Workspaces and Windows. Only one of the created Boards is able to be displayed at a time, while it is offered a context sensitive Menu that displays different menu items depending on the component that it is launched (i.e. Board, Workspace or Window).

General Functionality

The educator during the lecture has the opportunity to keep handwrite notes on the board (as one would do on a common whiteboard) and also annotate the content of the board (e.g. annotate on top of two or more application windows). Additionally, during the lecture the educator is able to select and share specific content with the students, for example capture a snapshot of a specific area on the board and forward it.

In order to address challenges A and C, given that the CognitOS Classboard is aware of the position of the user in front of it, the functionality of the application Windows was extended to include the "Summon" and "Follow-me" functions. The former forces the Windows to relocate to a position near the summoning user; that can be quite helpful for out of reach items, while it can also alleviate reachability constraints (e.g. younger students, disabled users using wheelchairs). Regarding the "Follow-me" function, it enables the educator to select a number of Windows that will follow their moves, making it easier to have constant access to specific applications without moving across the classroom to reach them. Finally, in order to address challenge G, the system has the ability to nudge a window or change the color of its border so as to draw the students' attention to it. All this functionality is offered either automatically by the system or through a sophisticated menu described in Section "CognitOS Classboard Menu".

In more detail the available functions are:

Follow-me. The users can select a specific Window to follow their moves (as visualized in Figure 24a). When the follow-me functionality is enabled, the Window is transferred into a new layer on top of the other Windows of the board. At the same time, the system monitors the movement of the user inside the physical environment (via the position tracking service) and relocates the Window accordingly. That way, the user has constant access to the window and can interact with its contents immediately. As soon as the user disables the follow-me functionality, the Window returns to its initial position.

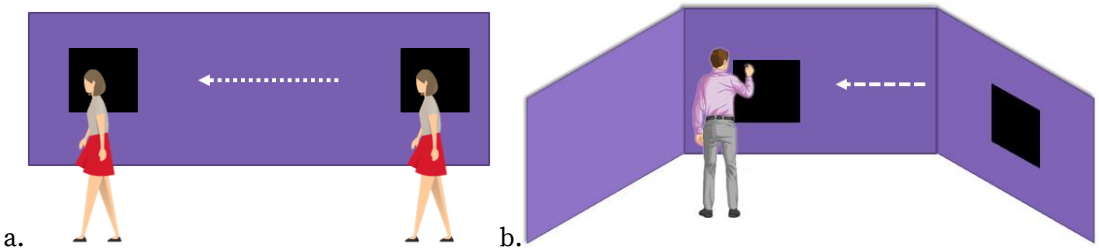


Figure 24: a. Follow-me, and b. Summon features.

Summon. The users can "summon" a selected Window near to their position (Figure 24b). In more detail, the users via the menu can launch the summon tool that displays a list of all application's windows existing in the active Board (Figure 25). From that list the users can select an application and "summon" it near them; that way users minimize their movements since they do not have to relocate near the desired application, while reachability constraints are also alleviated (e.g. height constrains, movement constrains, etc.).

Figure 25: Summoning an active window inside a workspace.

Reachability. This feature is provided automatically by the system. The board exploits the information coming from the reachability and position services. The combination of this information provides an indication of whether the content of the board should be rearranged (e.g. move the windows in a "lower" height). Thus, all the items will be reachable for example to the short users or to disabled users using wheelchairs.

Draw attention. The board has the ability to nudge a window or change the color of its frame so as to help the students to focus on it (Figure 26).

Figure 26: Draw attention feature.

Annotate. The user can activate or deactivate (launch or close the application) the annotation functionality via the menu. When the user activates annotation, a new transparent layer over the items of the board is enabled. The user can keep handwritten notes and highlight content (as it is visualized in Figure 27). Annotation is not application specific (e.g. the educator can highlight the content of two applications application Windows in a single run). All this content is kept on the annotation layer and remains on the board unless the user clears them.

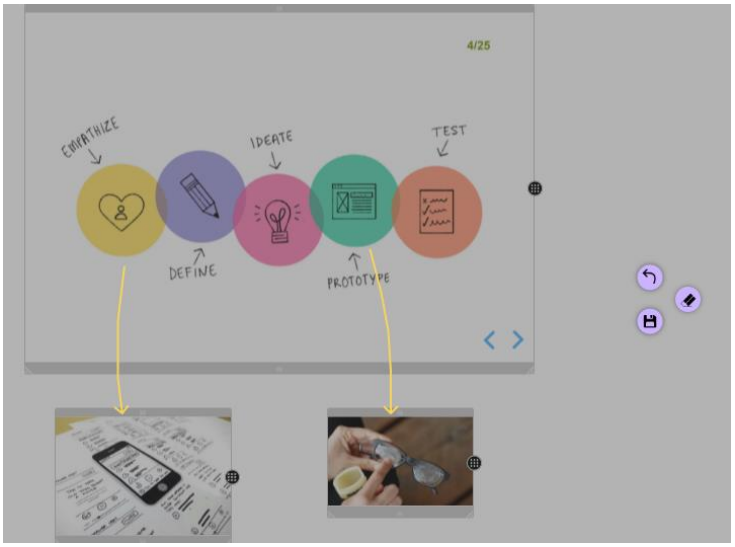


Figure 27: Annotation functionality.

Capture. When the screen capture is enabled, a new layer over the entire board is created, permitting the user to select the area that to be captured (as it is visualized in Figure 28). It permits educators and students to capture an

image of an entire workspace or a specific part of it (e.g. the user enables the screen capturer application via the menu and draws a diagonal line defining a rectangle whose contents are then captured). At the end, the capture is loaded on the board and the educator can interact with it (e.g. share it with the students). All the screen captures are stored by the system and can be displayed via the "Image Viewer" application.

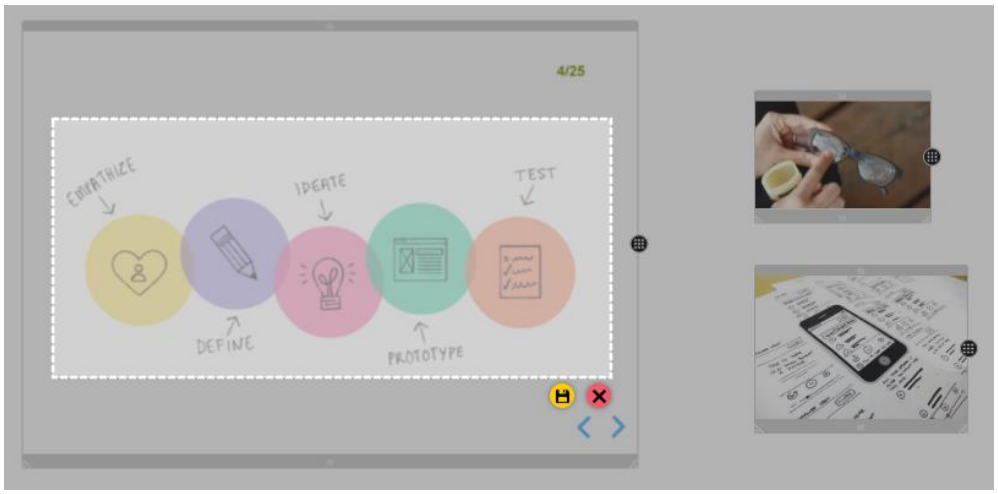


Figure 28: Capture functionality.

Record. Gives the opportunity to the educator to record the contents of the entire board. This application records all the actions that take place on the board; simultaneously, it synchronizes the captured video and audio properly. Additionally, it gives the opportunity to the users to pause, resume, stop recording and save the captured video at any time. The captured videos are stored in the Datastore of the system (descripted in the section 6.3.4) and can displayed by the "Video player" application.

CognitOS Classboard Menu. The CognitOS Classboard offers a context-sensitive Menu that follows a modular design. The contained functionality is organized under four categories:

i) Board: includes functions that apply to Boards

ii) Workspace: includes functions that apply to Workspaces

iii) Tools: includes global functions

iv) Window: includes functions that apply to Windows

v) Application: includes functions that are Application specific.

These categories are either expanded (to reveal their functions) or collapsed depending on the component on top of which the Menu is launched (i.e. Board, Workspace, and Window). In more detail, if the user opens the menu over an empty area of the Board the menu is adjusted in "Board" mode, over a Workspace it is adjusted in "Workspace" mode, over a Window residing on the Board it is adjusted in "Window in Board" mode, and finally over a Window residing in a Workspace, it is adjusted in "Window in Workspace" mode (Table 2).

Table 2: For each menu's mode which components are visible.

Modes	**Menu Categories**				
	Applications	Board	Tools	Window	Workspace
Board	√	√	√		
Workspace	√	√	√		√
Window in board		√	√	√	
Window in workspace		√	√	√	√

To open the menu, the user can interact with the system on-surface or above-surface. More specifically, in the first case of the "on-surface" interaction the user can open the menu via right clicking or by making the appropriate gesture on the board. In this case the system recognizes the area of the interaction (e.g. board, workspace, window) and both computes the position to open the menu and adjusts it in the appropriate mode. In the second case, the user opens the menu via the tablet or the smart phone controller, by using the appropriate voice command (e.g. "menu open") or by using the appropriate mid-air gesture. In order to specify the position and the mode of the menu, the system uses the position tracking service (i.e. provided by classroom's environment) and combines the received data (i.e. educator's

position) with the state of the Classboard (i.e. position of opened applications or position of existing workspaces).

Board manager. It provides an overview of the board's state. The large board is packed in a smaller Window (as visualized in Figure 29) and permits users to manipulate the state of the board in an easy manner. This approach gives the ability to the user to manage the state of the board with less effort, by spending less time and by making fewer body moves than usual.

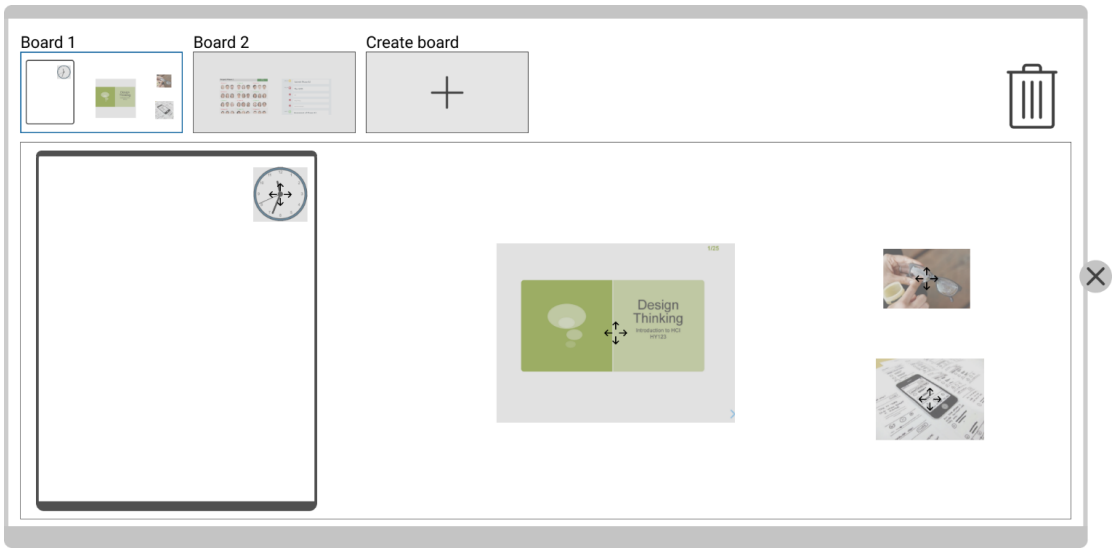


Figure 29: The board manager.

In more detail, the board manager application is accessible via the tools menu; when the user opens the application, the system place application's window in a higher-layer over the main content of the board. The application's interface is split into four main parts as follows:

- In the center area of the window, the "board" is visualized at a small scale, but by providing the same functionality as the real board. The user can interact with the content (e.g. rearrange or resize the windows, select windows and group them in new workspaces), and those changes simultaneously update the state of the "board".
- The top area of the window displays a navigation menu, through which menu the user can load a different "Board" from the existing or creates a new one. By dragging and dropping a Board over another Board, those two could merge (the applications of the first one move in the second).

- The bottom side of the window visualizes the content of the workspaces. When the user selects a specific workspace, it is previewed in this area. The user can interact with the content of the workspace and create new alternative views of the workspace.
- The left side of the window contains some action's buttons and bin's area. More specifically, the user can undo and redo an action or restore the board in its initial state (the state of the board when board's manager application launched). Finally, the user can close a window or remove a workspace by dragging and dropping the specific item in the bin's area.

All of the actions that are dynamically accomplished via the manager update the state of the board at run time.

Immersive Experiences & X-Reality Technologies

In the Intelligent Classroom of ICS-FORTH the role of the walls is dual; on the one hand, they act as interactive smart boards, where typical educational content (e.g., multimedia, notes, exercises) can be presented, and on the other hand they are able to create simulations that fill the user's visual field, creating a perception of physical presence [88]. Such simulations, namely immersive environments, are beneficial for students, since they enable interaction with the real world in ways that were not possible before and create new situations that would be impossible to create solely in a traditional or in a digital setting. The CognitOS Classboard has the ability to create a CAVE-like stereoscopic environment by displaying attractive, situational (i.e. course-, topic- and discussion-specific) multimedia on the Intelligent Classroom Board (Figure 30). That way, students can get the feeling of being present into any environment relevant to the course's syllabus (e.g. a cave or a rainforest). Additionally, the system interoperates with the core Ambient Intelligence facilities of the Classroom [60] in order to create appropriate conditions (e.g. light, sound, scent) and enhance the feeling of immersion. For example, consider the following setup that can immerse students into a rainforest: a video displaying a forest covered chiefly with trees sprouting in a riverbed,

dimmed classroom lights with green tint and an ambient soundscape featuring water and bird sounds.

Figure 30: 3D representation of the CognitOS Classboard immersing students into the Dordogne caves.

Additionally, advanced X-Reality technologies are employed to promote the way students learn. In conjunction with the immersive environment of the classroom, these technologies are used to create highly compelling experiences, through which students can improve and explore different skills and practices. In more detail, using their handheld devices (e.g. tablets or smartphones) or wearing special glasses (e.g. VR glasses), educators and students can access Augmented Reality (AR) applications. Augmented reality is an interactive experience where objects that reside in the real world are enhanced by computer-generated information. When used in educational settings, AR promotes enhanced learning achievements [89], [90]. The CognitOS Classboard features sophisticated mechanisms that permit such AR applications to identify objects of interest (e.g. a specific tree in the forest), by projecting subtle cues on the wall, which act as markers enabling the virtual enhancement of the displayed world. Consider the following example: during an immersive course on rainforests, where the classroom setup has been transformed accordingly and the walls display videos of raindrop covered trees glistening in the sunlight, students can utilize the AR applications of their tablets to search around for hidden clues. These clues unfold further information, such as details on the forest's flora and fauna.

In order to support the educator during a lecture and enhance learning, a suite of educational applications along with general purpose tools and utilities were developed.

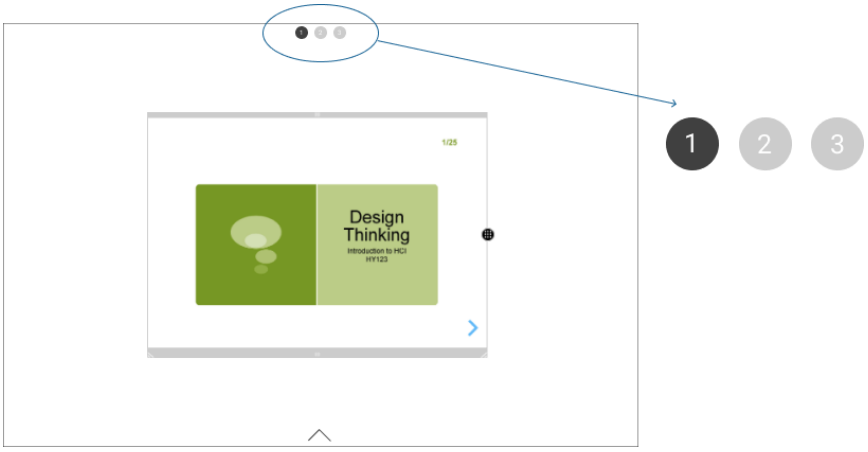


Figure 31: Paging component.

5.3.2 Board Component

The educator can create a new Board on-the-fly or open a previously defined one. The ability to define a Board a-priori partly addresses challenge C, since it helps educators in preparing their course offline and save time from launching and positioning the desired applications during the limited class time. The CognitOS Classboard permits educators to create multiple Boards to allow a better organization of application Windows. For example, an educator might select to use a Board for displaying various utilities relevant to the course (i.e. a calculator) and another Board to present multimedia content. Educators can switch between active Boards in a linear manner (e.g. from the first to the second, etc.) or select a specific one. Additionally, two or more Boards have the ability to merge on demand by moving all their applications to a single one. At the bottom of a Board there is a dedicated touch-enabled area, were the educator can perform a swipe up gesture in order to reveal the next Board. On the top area of the Board, there is an interactive control (in the form of numeric pagination) that indicates the number of existing Boards. That control is used to provide information about the active Board (e.g. the 2nd Board out of three existing Boards is visible), while the users can load a specific

Board by simply clicking on the appropriate number of the paging component (Figure 31).

Each Board can host multiple educational applications, while an underlying intelligent mechanism can automatically arrange their position or toggle their visibility based on numerous parameters (e.g. available space, educator preferences, past knowledge), if required (e.g. there is not enough space). The functionality available through the Board-specific Menu is described in Table 3 (visualized in Figure 32).

Table 3: Functionality of CognitOS Classboard Boards.

Functionality	*Description*
Top Board Area – Pagination Component	
Change Board	Makes another Board visible, in case there are alternative Boards available
Bottom Board Area	
Change Board	Makes another Board visible in a linear manner, in case there are alternative Boards available
Board specific Menu	
Menu-Board	
Next	Navigates to the next alternative Board
Previous	Navigates to the previous alternative Board
Create New	Creates an alternative Board
Menu-Tools	
Summon	Relocates an existing Window to that particular Board
Annotate	Enables or disables annotation application
Capture	Captures the contents of a selected area
Board manager	Launches the Board manager application
Record	Starts / stops board's screen recording
Menu-Applications	
Most used Applications	Launches a selected Application from a list of the four most used Applications, or expands the list of the available Applications

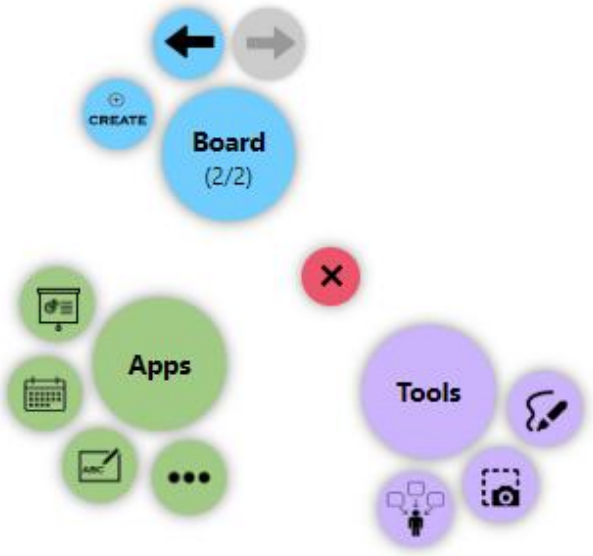


Figure 32: Board-specific menu.

Table 4: Functionality of CognitOS Classboard Workspaces

Functionality	*Description*
Workspace frame	
Resize	Changes the dimensions of the Workspace
Move	Changes the position of the Workspace
Pagination Component	
Change Workspace	Makes another Workspace visible, in case there are alternative Workspaces available
Create New	Creates an alternative Workspace of the same dimensions
Workspace specific Menu	
Menu-Workspace	
Next	Navigates to the next alternative Workspace
Previous	Navigates to the previous alternative Workspace
Menu-Tools	
Summon	Relocates an existing Window to that particular Workspace
Annotate	Enables or disables annotation application
Capture	Captures the contents of a selected area

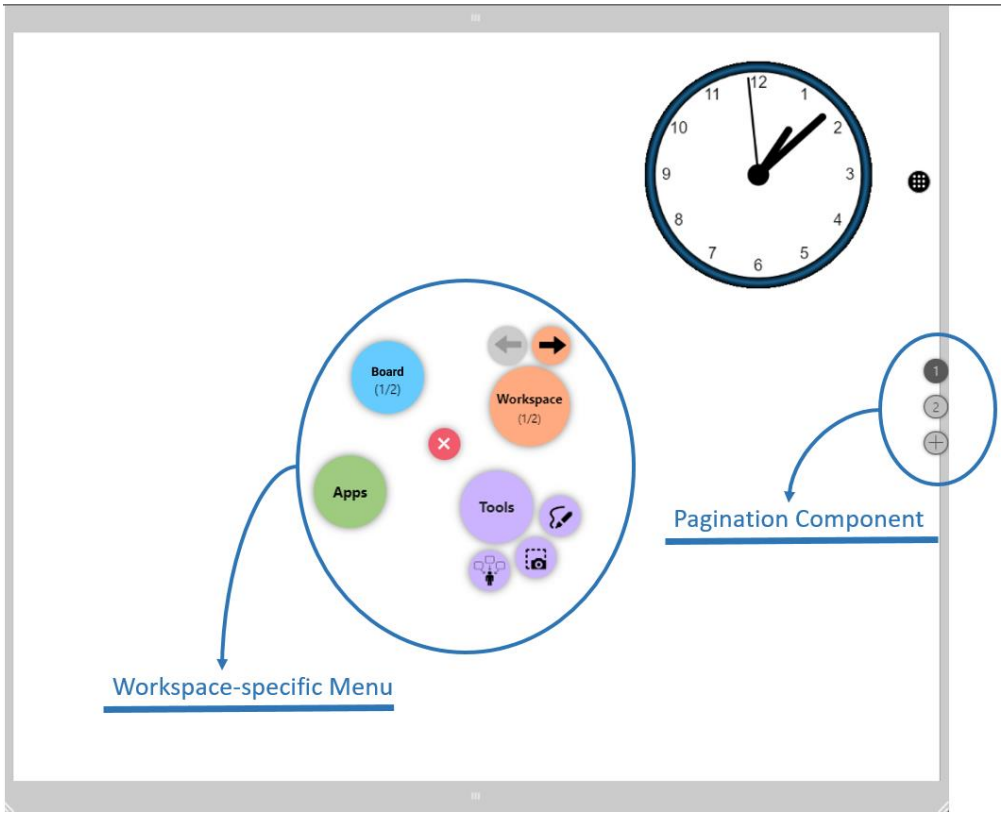


Figure 33: Workspace-specific Menu.

5.3.3 Workspace Component

The educator has the ability to group two or more application Windows and organize them in a Workspace. As depicted in Figure 33, the workspace component is framed; this framed area is used to move and resize that item. When the user clicks on an empty area of the Workspace, a Workspace-specific menu appears, containing the functionality which is presented in Table 4. The educator can create several Workspaces on a Board, given that they fit to the available space. For example, a Board can contain two large Workspaces covering the 100% of its dimensions, or several smaller Workspaces. Additionally, for each Workspace educators can create up to four alternative views (soft-limit). In more detail, behind each visible Workspace there is the opportunity to create other Workspaces of the exact same dimensions containing different application Windows. The educator has the opportunity to alternate amongst these Workspaces and reveal new content to a specific location on the Board. For example, consider an educator giving a lecture regarding the Battle of Thermopylae; the educator has always

visible the PowerPoint file on the right side of the Board and alternates amongst two parallel workspaces -placed on the left side of the Board- containing the Multimedia application window and the Map application window respectively (Figure 23). The frame of each Workspace contains a pagination component permitting navigation amongst the alternative Workspaces. The entire functionality available through the Workspace component is described in Table 4.

5.3.4 Window Component

Windows can host either built-in or external applications, which are described in section 5.5. In practice, it is a frame that can embed a viewer to any existing web application (e.g. applications that accessed via http) or webpage (e.g. google search, google maps, etc.).

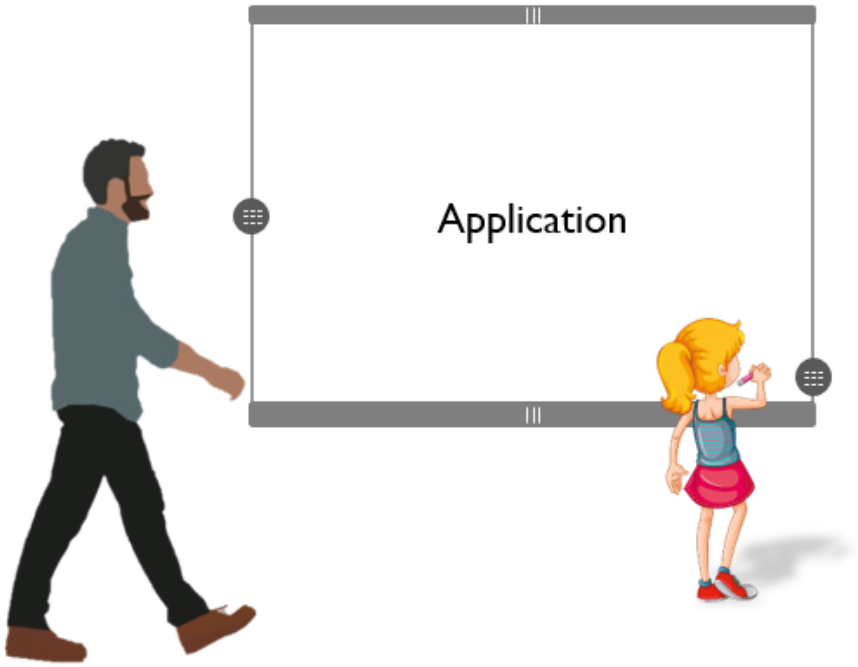


Figure 34: Reachable interactive components.

The CognitOS Classboard Windows have a thin frame that is used to move and resize them; the main color of the frame has a subtle greyish tint (visualized in Figure 34), which however is able to change in order to draw the students' attention to a specific Window (see Section "Draw attention"). Furthermore, the Window frame contains a menu button, which is used to open a context

sensitive menu. The positioning of the button depends on the location of the user who interacts with it; for example, if the user approaches the right side of the Window, the button appears on the right of the frame, while it is vertically aligned in the middle of the frame so as to ensure reachability (visualized in Figure 34).

Additionally, educators and students can manually perform a variety of actions on the application Windows, including: closing, maximizing, resizing, rearranging (in the current Workspace or between Workspaces), pining, show/hide their frame (Figure 35) and grouping. The pin functionality refers to the ability of setting a fixed position for an application window within a specific Board or Workspace. The entire functionality available through the Window component is described in Table 5.

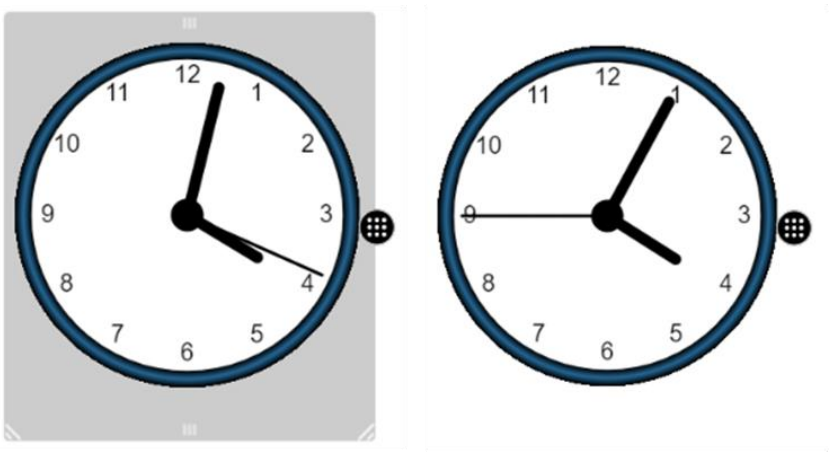


Figure 35: Show / hide frame feature.

Table 5: Functionality of CognitOS Classboard Windows

Functionality	*Description*
Window frame	
Resize	Changes the dimensions of the Window
Move	Changes the position of the Window
Change Color	The color of the Window frame changes automatically, in order to draw attention to it
Window specific Menu	
Menu-Application Name	
Close	Closes the Window

Pin	Pins the Window to a specific location
Maximize	Maximizes the Window
Show/Hide Frame	Shows or Hides the Frame of the Window
Follow-me	Enables or disables Follow-me functionality for that Window
Application specific actions	Performs actions exposed via each specific application
Menu-Tools	
Annotate	Enables or disables annotation application
Capture	Captures the contents of a selected area

Figure 36: Window-specific Menu (Window residing on the Board).

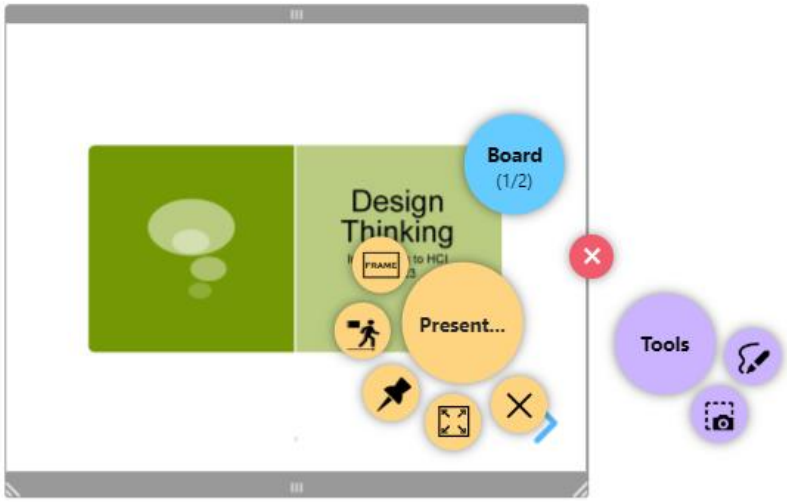


Figure 37:Window-specific Menu (Window residing in a Workspace).

5.4 Interaction Paradigm

As a result of extensive research on interaction techniques, CognitOS Classboard was designed featuring multimodal interaction methods in order

to fully address challenges A, C and D, including multi-touch, touch gestures, mouse gestures, mid-air gestures, voice commands, and user position tracking. The interaction with the various aspects of the CognitOS Classboard (i.e. Boards, Workspaces and Windows) can be divided in two categories: on-surface interaction and above-surface – or air-based – interaction.

5.4.1 On-surface Interaction

Interaction on the surface signifies touch interactions directly on the reachable parts of the display using fingers, hands and tangible objects, in order to select, grab, throw, scale, move objects, etc. The CognitOS Classboard supports on-surface interaction both through the interactive marker and the user's fingers/hands. In more detail, the system can identify fine movements and differences between the number of fingers and hands used at each moment, thus enabling the creation of an extensive library of various gestures. Various on-surface gestures are employed in order to speed-up interaction with windows and workspaces (i.e. pan left to move a window towards that direction), while more advanced gestures (e.g. circles, numbers and letters) can be used as time-saving shortcuts. Such shortcuts can trigger predefined system actions like opening a menu, alternating between workspaces, summoning applications, etc.

Figure 38: 3D representation of the CognitOS Classboard displaying educational applications and a picture taken from in the actual Intelligent Classroom setup.

5.4.2 Above-surface Interaction

While interacting with large displays, artifacts may be out of reach and selecting items may require excessive movement [87], while when touching the screen, the user can only see the area directly in front of them. In order to overcome these difficulties, the system detects gestures and directions indicated by the movement of users (e.g. hands, arms, body) in order to support interaction while being far from the screen (e.g. change workspace, summon an application). Hence, the CognitOS Classboard features a set of mid-air gestures that correspond to simple or complex actions (e.g. swipe left to go to the next slide, clap twice to clean up the entire workspace). Furthermore, implicit interaction is achieved by tracking the position of users in front of the board and reacting accordingly (e.g. rearranging windows, delivering personalized content). Another mechanism can infer the role of the user (i.e. educator or student), making it possible to address challenges E and F. Finally, a small set of voice commands are available for educators; for example, the utterances “next workspace” and “previous workspace” permit educators to switch between active workspaces in a linear manner.

5.4.3 Remote Interaction from Desktop & Mobile platforms

Given the size of the projected area (the wall in front of the entire class and one side wall), it would be cumbersome for educators to keep track of every application launched on the board and its status. Additionally, keeping in mind that educators often spend time sitting at their desk, expecting them to stand up and move in front of the board to perform actions that could be easily done from their workstation (e.g. launching a multimedia viewer) seems unrealistic and inefficient. To this end, a smartphone, a tablet and a desktop application (Figure 40) were developed enabling educators to get an overview of the current board status (this tackles challenge B), manage the board Workspaces and manipulate application Windows. For example, during a class the educator can use the tablet to rearrange the Windows of the active Board, use the smartphone application to perform actions over the Board or

specific applications, while when at home, they can use the desktop application to easily pre-define Workspaces for future courses.

Desktop Application

The CognitOS Classboard provides a desktop application (Figure 39) to configure the state of the board a-priori. Using this application, the educator can prepare the layout and the content of the board before the lecture. In more detail, the educator (i) can create new set-ups for the Classboard (as set-up is the configuration of the board's layout and content), and (ii) has access to the already defined set-ups in order to edit them (e.g. update, delete them, etc.).

Using the desktop application, the educator can create a new set-up, which will be presented in the appropriate lecture. The process is split in two parts. In the first part the educator fills in some general information about the specific set-up, while in the second part he/she prepares the content of the board. Therefore, the educator at the beginning must provide a name to the specific configuration and then a short description; such information will be useful in the future to help the educator to recognize and retrieve a set-up. Additionally, some extra information is required regarding the course that is being set-up (i.e. name, date and time).

In the next step, the educator has to configure the layout and the content of the board. Therefore, the educator can create new Boards, and organize them. On each of the created Boards, the educator can launch applications and place them into specific positions, while at the same time he/she can resize, pin, maximize them and initialize their content (e.g. open the presentation viewer and load a specific presentation file). Furthermore, the application Windows can be grouped into Workspaces, which can be also manipulated the same way as Windows (e.g. move, resize). As soon as a set-up is final, the educator is able to save it and load it to the board when needed. The educator has access to a personal library that contains all the set-ups configured by him/her; from that list the educator can update or delete any of the existing set-ups.

Finally, it is important to mention that the application provides an assistant, which provides helpful tips during the aforementioned process (e.g. how to set-up the layout of the board, if some applications should be grouped in a workspace, suggests positions to place the items, etc.).

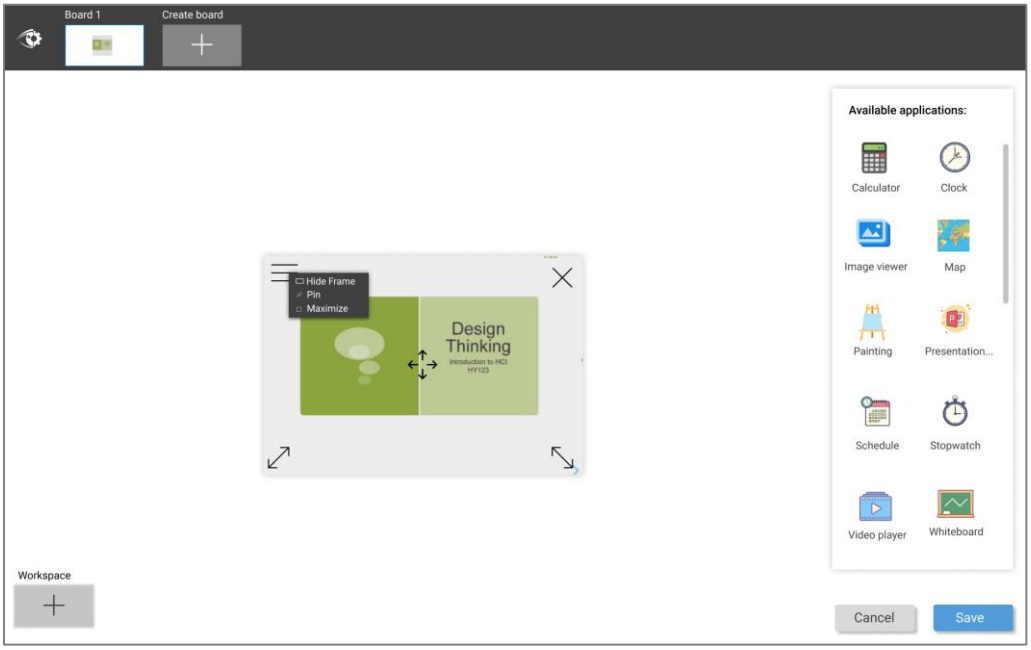


Figure 39: The desktop application.

Tablet Application

The CognitOS Classboard provides a tablet application (Figure 40), installed in the educator's workstation (section 4.2.3) which helps the educator to have an overview of the board's state during the lecture. Using this application, the educator can manage remotely the state of the board and manipulate its content.

In more detail, the educator can change the active / visible Board by selecting another existing Board or creating a new one. Furthermore, he/she can rearrange or resize the visible Board items (i.e. Workspaces and Windows), group applications Windows in new Workspaces, pin Windows in specific positions on the Board, enable follow-me functionality for specific Windows, etc. At the same time, via the tablet application, the educator can launch new applications and place them on the active Board.

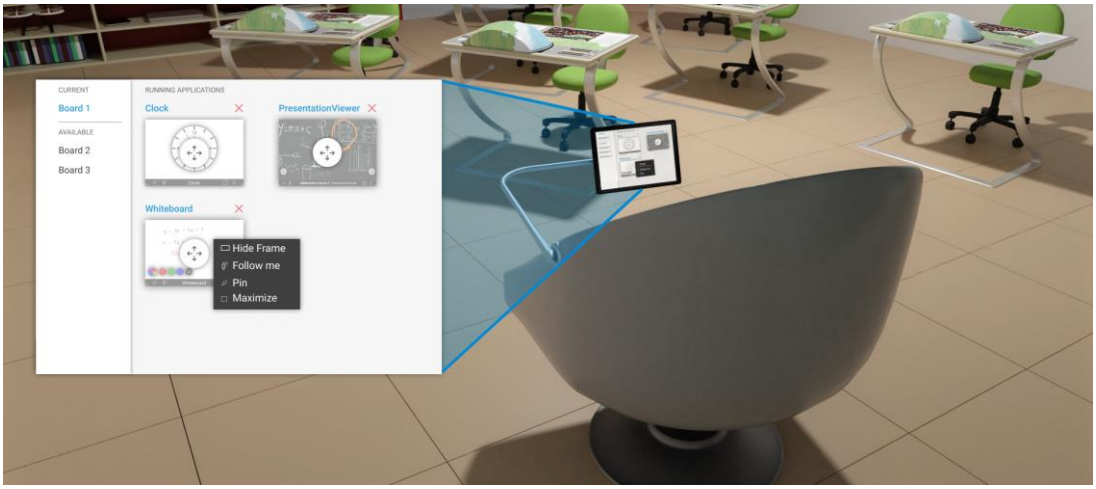


Figure 40: The tablet application running on the educator's workstation.

Smartphone Application

The CognitOS Classboard provides also a Smartphone application (Figure 41). This application can be used by the educator as a mini remote controller. Using this controller, the educator is able to interact with the board remotely and update its state. More specifically, the educator can change the active Board or Workspace by navigating to the next or the previous or selecting a specific one. Furthermore, via the Smartphone controller the user can interact with the launched applications of the active Board. In particular, the user can close, pin, unpin an application Window, while at the same time perform various application specific tasks (as described in section 6.4.1).

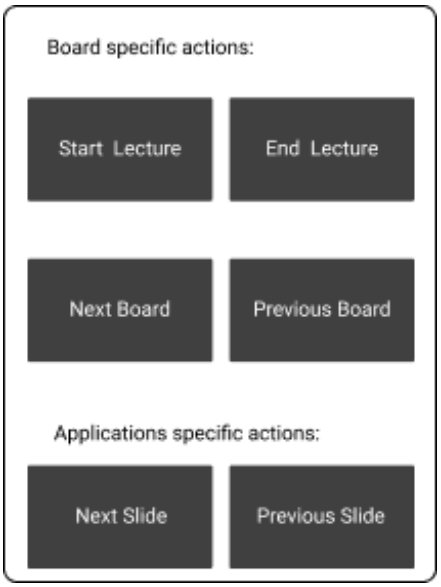


Figure 41: The smartphone application.

5.5 Applications Suite

The CognitOS Classboard, in order to support educators during a lecture and enhance the learning process, provides an application library which consists of a number of educational applications and other useful tools. More specifically, the applications suite is divided into two categories, (i) built in applications and utilities and (ii) external applications. Built-in applications were designed and developed for CognitOS Classboard, while external applications were imported by following some specifications described in section 6.4.1.

5.5.1 Educational Applications

Built-in and external educational applications.

- The **Presentation viewer** (Figure 42a) enables educators to present, highlight and annotate course slides. The educators prepared the slides a-priori and present them during the lecture.
- The **Image Viewer** (Figure 42b) displays image content. The educator can create collections of photos and show them to the students (e.g. preload photos to the application, create collections of photos during lecture). The image viewer application gives the opportunity to the educator to share the displayed content with the students.
- The **Video Player** (Figure 42c) as implied by its name, displays video files. The educator could load videos and show them to the students during the lesson.
- The **Exercise viewer** permits the online representation of various types of exercises (e.g. multiple-choice, fill in the blank).
- The **Whiteboard** (external) application permits users to sketch and handwrite notes on the wall (as one would do on a common whiteboard).

a. 

b. 
c. 

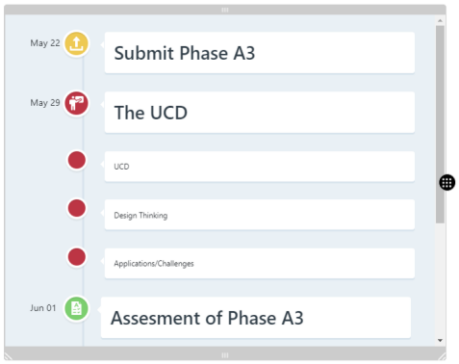
Figure 42: (a) Presentation viewer, (b) Image viewer, and (c) Video player.

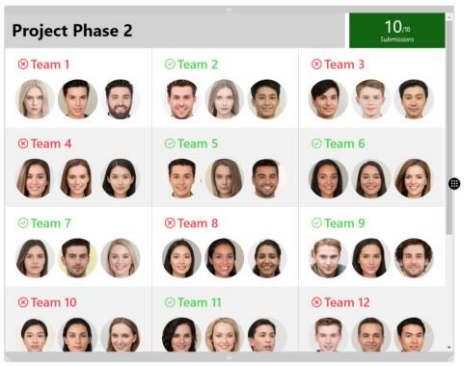
5.5.2 Utilities

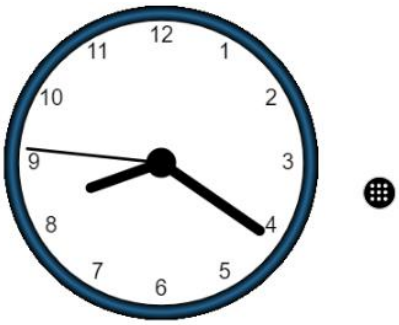
- The **Calendar** application (Figure 43a) presents the daily schedule of the classroom, as well as reminders for future events (e.g. quizzes, presentations).
- The **Calculator** (external) application permits the execution of advanced mathematic calculations.
- The **Clock** application (Figure 43c) supports alarms, timers, and a stopwatch (Figure 43d). It can be customized to use different styles (e.g. analog or digital) and colors.
- The **Exercises submission** application (Figure 43b) as it implied by its name is used by the students to submit their exercises. Simultaneously, the students could use the application in order to be organized in working groups. Additionally, provides a submissions' viewer to the teacher, in order to have an overview of the submitted exercises.
- The **Geometry assistant** (external) identifies handwritten geometry shapes on the board (e.g. triangles, circles, etc.) and transforms them into high-level, computer generated shapes.
- The **Graph assistant** (external) offers similar functionality to the Geometry assistant, identifying handwritten graphs.
- The **Map** (external) application features an interactive world map which can be zoomed in at specific places. By selecting a point (e.g. a city, or a mountain) the user can view additional information. Using AR

technologies, the various places of the map are presented in 3D to give a better sense of space. It opens by default during the Geography course.

- The **Notes** application can be pinned on top of other applications and permits educators to create handwritten messages that are always visible.
- The **Periodic table** (external) contains interactive chemical elements, which can be highlighted and grouped in order to create chemical compounds. It opens by default during the Chemistry course.

a. 

b. 

c. 

d. 

Figure 43: a. Calendar application, b. Submissions application, c. Clock application analogue mode, d. Clock application stopwatch mode.

Chapter 6

CognitOS Classboard Implementation

The CognitOS Classboard framework is built based on a micro service architecture style [66]. It is composed of a number of loosely coupled services, which are independently deployable, maintainable and testable, thus enabling the system to be flexible and scalable. Every service is standalone, it encapsulates the complexity of its mechanisms and exposes a public API to be consumed by other services (e.g. the services of the classroom's environment). In practice, the CognitOS Classboard framework comprises a different number of light-weight components that are combined to create the core modules that exists in one of the following categories:

(i) CognitOS Classboard Core

(ii) User Interface Framework

(iii) CognitOS Classboard Utilities

(iv) Applications suite.

Furthermore, CognitOS Classboard exploits a number of services which are provided by the AmI environment of the classroom; those services are

grouped and described as “Classroom’s services”. The aforementioned components interact and exchange data (as it is visualized in the Figure 44) so as to ensure that CognitOS Classboard functions properly. At the center of the architecture, the core component which is responsible to orchestrate all the available components and their services is visualized. The core component is responsible for handling the incoming and the outgoing requests related to the applications suite, the user interfaces, the utilities and the Classroom’s services (the systems described in Chapter 4, namely CognitOS, AmI-Solertis, Lector and Classmate, as well as the other artifacts of the classroom e.g. Intelligent student desk, educator workstation, etc.). In more details, it requests and exploits information from the other components and at the same time it exposes a public API, thus permitting them to interact with the CognitOS Classboard and update its state.

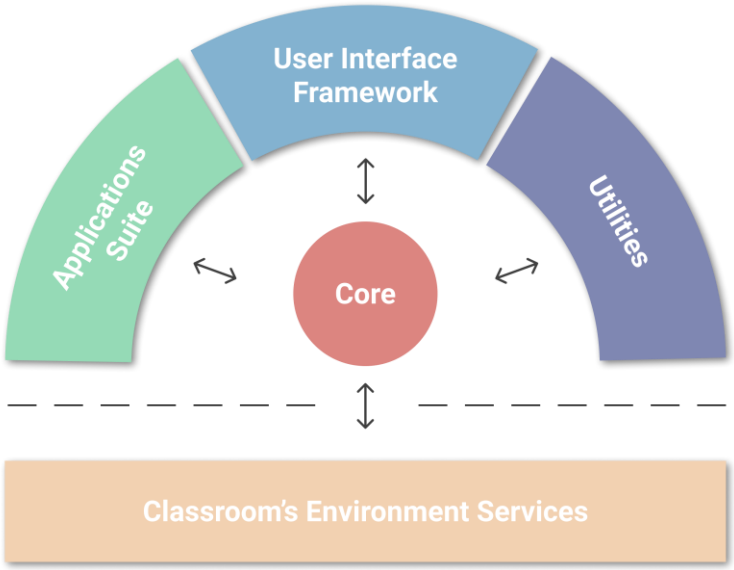


Figure 44: The four basic modules that constitue CognitOS Classboard framework and the Classroom’s enviroment services module.

The user applications suite component provides to the framework educational applications and tools in order to use them and enhance the lecturing process. The provided applications are launched, hosted and controlled by the services that constitute the user interfaces component; additionally, it provides tools to handle the state of the Classboard. It is very important to secure the system and avoid failures, crashes and problematic situations, and at the same time

to provide access to a central data store and other useful tools; this functionality is grouped in the utilities component and provided to the system. Finally, in order to exploit the available services provided by the Classroom's environment, a number of bridges services are developed (which encapsulate the needed functionality from the environment's services) and grouped in the Classroom's services library.

6.1 CognitOS Classboard Core

CognitOS Classboard offers a set of fundaments services that play a vital role in the framework's functionality and are encapsulated in the core component. Those services are responsible to handle operations related to the state of the framework, the management of the items on the board by applying the implemented items placement algorithm, the interaction of users with the system, etc.

6.1.1 Classboard controller

The Classboard controller service is part of the core component. As it is implied by its name it controls the Classboard; more specifically, it is the main controller service of the system. The service is responsible for handling the requests and the events that are related to the Classboard. More specifically, as the main controller, it handles incoming and outgoing requests and events that are related to the Classroom's environment services, the user interfaces, the applications suite and the utilities.

In more detail, the controller receives information from the various systems that monitor the environment (e.g. Lector, AmI-Solertis, Classmate) and handles requests coming from CognitOS, students' desks, educator's workstation. Additionally, it retrieves information from the Classroom's services related to: (i) the state of the classroom's environment (e.g. temperature, lights, etc.), (ii) each user in the classroom (e.g. information about the user's profile, which is the role of that user, etc.), and (iii) interaction information that is recognized via the environment (i.e. voice commands,

gestures, position tracking data), etc. Simultaneously, it outputs to the environment data regarding the state of the Classboard (e.g. open / close, application specific information such as the "presentation has started"). Furthermore, it informs the environment about events that it is interested in, for example that it is interested in specific gestures or voice commands, the position of specific users, etc.

The controller handles information from the applications suite. In more detail, it gets information about the available applications, about the general-purpose tools and other utilities applications. The controller forwards that information to the user interface application so as to be used and exploited by the users. At the same time, the controller provides information to the suite related to application usage statistics, information about malicious or problematic applications, and other.

Furthermore, the controller acts as a content provider as well. Classboard, desktop, tablet, and mobile applications request configuration information and extra data about the state of the board, the available applications, etc. Conversely, those applications provide to the controller information related to updates of the board's state, control requests coming from mobile, tablet applications and setup information coming from desktop application.

Moreover, the controller provides information to the utility's services and vice versa. In more detail, the state monitoring service requests information related to the state of the used services, the applications provided via the applications suite, etc. On the other hand, the controller requests from these services information about the overall state of the system.

Finally, the controller handles requests and events coming from the interaction related services, such as on-surface, above-surface and remote controllers, shortcuts manager, reachability recognition service, etc., and adapts the functionality accordingly.

6.1.2 Logical board manager

As the manager of the board, this service is responsible for handling all the actions that are related to the board component. The service has always the overview of the board's state and it handles all the requests and the published events (in the event federator) that are related to the board. In more detail, each time it receives a new request or event, it decides how to act based on the state of board and other contextual parameters and either forwards it to the appropriate board's related service immediately (e.g. if the board's services are available, or if the action is critical and must be executed as soon as possible), or schedules it for later, or ignores it. The board decides that the requested action should not be executed, it is meaningless or out of context; e.g., discard a "move to the next board" action if only a single board exists. In general, the service evaluates requests and events which are related to actions that update the state of the board, for example the opening of a new application, the movement of windows, request to move in the previous or next Board or Workspace, shortcuts, etc. As already mentioned, the logical board manager handles interaction related event data that are produced by the users, receives recognized gestures, voice commands and instructs the board's services how to execute specific actions. In more detail, when the manager receives interaction related events such as the gesture "w" or the vocal command "next slide", it interprets the request and informs the corresponding services accordingly. Specifically, if the request is related to the opening of new applications, the Logical board manager collects and provide to the items usher the necessary data. Additionally, based on environment data (i.e. the user's position), the state of the board and historical data, it decides where the items should be placed and which configuration policy should be followed.

6.1.3 UI Components Usher

One of the basic features of the UI Components Usher service is to map a newly created item to a place on a given area. More specifically, the manager gets as input an item (i.e. a window or a workspace), the area where it will be placed

(i.e. Board, Workspace) and some extra configuration parameters of the service; it processes all those data, finds and returns as an answer the position where the item will be placed. In order to understand the nature of the problem and implement the items placement algorithm of CognitOS Classboard, a study was conducted in the domain of window placement algorithms and more general in the domain of bin packing algorithms. During this study a number of notable research works were found.

The main point of the existing windows placement algorithms is to look for an appropriate place for the new windows. In [94], an algorithm is proposed which make use of the state of the workspace and tries to find a position for the new window causing the least overlap with the existing windows. In [95] the window manager "sawfish" was proposed, with a very minimal policy compared to most window managers. Actually, its aim is to manage windows in the most flexible and attractive manner possible. The window placement algorithm of this window manager supports multiple ways of placing new windows on the display; a number of different variables could be configured in order to apply a different placement method. The most important variable is the "placement-mode". This manager supports a number of different modes such as place a window randomly, place the window in the center of the screen, allow user to place the window in a preferred position, place the window in the first available position, look for positions where the new window would have a small overlap, etc. One of the major distinctions between the existing approaches and the approach of the CognitOS Classboard is the overlap of the windows, which in the case of CognitOS is not allowed.

Based on the policy that the overlaps are not allowed, the study of window placement problem in-depth led the research in the area of the bin packing problem and the proposed algorithms in that area. Supposing that the algorithm starts from one single bin (e.g. only one board) and in that bin should be placed a number of items with different dimensions; if the algorithm cannot find a position in that bin, then it creates a new one and tries again. The mentioned problem seems to be similar to the bin packing

problem. As is defined in [96], in the bin packing problem "*items, of different volumes must be packed into a finite number of bins or containers each of a fixed given volume in a way that minimizes the number of bins used*". The bin packing problem is an NP-hard problem when formulated as an optimization problem and NP-complete as a decision problem; that means that there is no known polynomial time algorithm to provide a solution. In addition, many heuristic algorithms have been developed for example first fit, next fit, best fit, worst fit and many more. Furthermore, a large number of new algorithms and optimized versions of the existing has been proposed by the research community.

Describing the findings in more details, in the case of the **Next Fit** algorithm, if the new item fits in the same bin as the previous item, it is put there; otherwise, a new bin is opened and the item is placed in the newly created bin. It should be mentioned that this algorithm never goes back to a previous bin other than the bin where the last piece has been put. Another algorithm is the **First Fit**; in this algorithm the new item is put into the oldest (earliest opened) bin into which it fits. The only case to open a new bin is if the item does not fit in any bin that already exists. The next algorithm in the list is the **Best Fit** algorithm. In that algorithm, the idea is to place the next item in the tightest bin; that means to put the item into the bin in which the smallest space is left. Another bin-packing algorithm is the **Worst Fit**, whereby the next item is put in the emptiest bin among the existing bins and a new one is created only if the item does not fit into any other bin. In the case that two or more emptiest bins are found, then the item is put in the earliest created. Each one of the already mentioned methods has a decreasing variation. In more detail, the **Next Fit Decreasing** algorithm puts the items in decreasing order by size and then uses Next Fit on the resulting list. In the case of the **First Fit Decreasing** algorithm, again the items are put in decreasing order by size and then use First Fit on the resulting list. Another heuristic is the **Best Fit Decreasing** algorithm, whereby the algorithm firstly orders the items by size from the largest to the smallest and then runs the Best Fit algorithm. Last but not least,

the **Worst Fit Decreasing** algorithm sorts the items in decreasing order by size and in the next step the Worst Fit algorithm is used on the resulting list.
Based on the basic heuristic algorithms described above, a number of more efficient and effective solutions have been proposed. In the [97], the authors present lower bounds and dominance criterion and derive a reduction algorithm for the bin packing problem. On the other hand, [98] presents a new algorithm for optimal bin packing, rather than branching on the number of different possible bins that an item can be assigned to. Based on the bottom-left packing method and the next fit, first fit and best fit packing heuristics, in [99] develops and presents two new bin packing heuristic algorithms. Comparing these algorithms to other finite bin heuristics the authors conclude that the proposed algorithms are suitable for practical use. In [100] low-order polynomial time algorithms for near-optional solutions to the problem of bin packing are studied. By analyzing the First Fit and Best Fit the authors realized that these algorithms are members of a more general class of packing algorithms which have the same worst-case behavior. Making use of those results they present linear-time approximations to these class of algorithms whose behavior is as good as the behavior of First Fit algorithm under a large variety of restrictions on the input. In study [101] the authors present a general purpose hill-climbing method; after exhaustive research and a large number of comparisons between graph coloring, bin packing algorithms and the proposed method when applied on a wide variety of problems, they conclude that the hill-climbing approach can outperform the existing algorithms. One of the existing problems in bin packing application in 3-d space is that often a number of different criteria to be satisfied are conflicting (e.g. to minimize the bins used and to balance the load of each bin). In [102] an optimization algorithm called MOEPSO is presented. This algorithm after extensive investigations seems to be more effective and efficient compared to other algorithms for the same category of problems. In other more theoretical researches such as [103], the authors combine discrepancy theory techniques and a novel 2-stage packing algorithm and present an approximation

algorithm which is more efficient and matches certain combinational lower bounds. The study in [104] provides guidelines and general rules that should be followed in order to build a genetic algorithm for one set of NP-hard optimization problems, grouping problems. Other studies compare existing algorithms, heuristics and metaheuristics approaches. After the comparisons, the authors in [105] conclude that many instances of small to moderate size can be solved optimality, while larger instances can only be handled by approximation algorithms. Furthermore, in [106] the authors provide new worst case results for a number of classical heuristics such as first fit, best fit, first fit decreasing and best fit decreasing.

Items usher service

As already mentioned, the main task of the items usher service is to find a place for each newly created item on a given area. Based on the conducted review of previous studies as well as on the peculiarities of the CognitOS Classboard, a novel items placement algorithm has been elaborated. This algorithm is based on the main principles both of the windows managing and the bin packing algorithms. In more detail, the goal of each window placement algorithm is "*to look for a good place for the new windows on the screen*" and in the case of bin packing algorithms is "*to pack a collection of objects into the minimum number of fixed size bins*". It was recognized that those principles were vital to the needs of the CognitOS Classboard items placement algorithm approach and that it will be gainful if they are exploited. In order to give a solution to the problem of items placement, the pre-mentioned principles were adjusted to the context of Classboard. In practice, when a new item is created, the items usher service should look for a good position to place it (on the Board or on a workspace). "Good position" is defined as the one that satisfies criteria such as (i) find a place on the board as nearest to the user as possible, (ii) place the item near to other windows so as to leave smaller gaps and make good use of the available area, (iii) some applications remember their last position, so if it is possible place them in the same position; sometimes those criteria maybe have conflicts. Simultaneously, the algorithm

should handle the cases that there is no available space in the existing (Board or Workspace) and has to provide a solution to minimize the creation of new bins (i.e. to create a new Board or an alternative view of the workspace).

Based on those assumptions, a proprietary items placement algorithm is implemented by the items usher service and can be summarized as follows:

```
0.  Items Placement Algorithm
1.
2.  do {
3.    Check for available position
4.  } while (position found or area iterated )
5.
6.  if (position found) { return position }
7.
8.  create a new bin
9.  Find available position
10.
11. return position
```

The algorithm receives a number of input parameters, and in particular:

- **Placement area**. Information about the area in which a position for the new item should be found. In more detail, it is provided information about the state of that area (e.g. which items have already placed), extra configuration information for each one of the placed items (e.g. if the item is resizable, if the item is movable, what's the dimension limitations of the item).
- **New Item**. Information about the newly created item, for which a position in the placement area should be found (e.g. the preferred dimensions of the item, the dimension limitations, the preferred position or the last opened position).
- **Placement policy**. The algorithm supports a number of different modes to place a window, such as:
    a. Default: By default, the algorithm tries to find a position near to other items so as to leave smaller gaps and make a better management of the available space.

b. User's position: The algorithm tries to find a position near to the position of the user.
c. Last opened position: The algorithm tries to find a position based on the last opened position of this application.
d. User's preferred position: The algorithm finds a number of possible positions to place the item and provide them as suggestions to the user in order to choose the preferred position.

- **Items configuration policy.** The algorithm also supports a number of different configuration policies for the items in the board.
    a. Default: By default, the algorithm tries to place the item in a position on the board without making any changes to other items (e.g. move or resize them).
    b. Push items (only if it is allowed): If the algorithm could not find a position for the new item without making any change, it is allowed to push the existing items and place them in new positions in order to find available space. It is worth noting that the algorithm can push the existing items only if they can be moved (they are not pinned).
    c. Resize and Push (only if it is allowed): If the algorithm cannot find a position for the new item without making any change, it is allowed to push the existing items and resize them too. Resize and push changes can take place only if the items are movable (they are not pinned) and resizable (depending on dimension limitations).
    d. Resize and Push (check only dimension limitations): The algorithm tries to find a position without moving or resizing the existing item; if no available position is found, the algorithm tries to resize and push the existing items. In this case the only limitation is the dimension limits of the applications; even if an item is pinned, it can be modified in this case.
- **General Information.** Some extra information is provided such as the position of the user in the room, the exact place on the screen where the user interacts to open the application (if the request was sent via on surface

interaction). This information is exploited in cases that the policy is meaningless (e.g. in an empty board and in the case of default placement policy it is better to open the window near the user's position than in a random position).

Describing the algorithm in steps:
Step 1: The algorithm based on the provided placement policy and the default items configuration policy tries to find the first available position.
Step 2: If a position is found, the algorithm returns it; else it continues
Step 3: It tries to find the first available position based on the provided placement policy and the items configuration policy.
Step 4: If a position is found, the algorithm returns it; else it continues
Step 5: Creates a new "bin" and finds a position based on the placement policy. If the policy is inappropriate, then it makes use of the general information.

In more detail, when the service receives a new request for item placement, the main goal of is to find a position in the given area based on the given parameters. It is worth noting that the algorithm does not allow items to overlap and that in default cases it tries to find a position for the new item that does not create large gaps between the new and the existing items and tries to place the item without any changes over other items. Additionally, it should be taken into account that the algorithm always returns the first position that satisfies the provided policies; in this phase no optimization is made (the algorithm does not try to find the best available position). In the first step, the algorithm tries to found an available position based on the provided policy; it iterates the given placement area until it has found the first available position. In that step the algorithm does not make any changes to the existing items, it simply applies the default items configuration policy. This step is completed when the algorithm finds the first available position or if all the placement area is iterated. After the completion of the first step. if a position was found the process stops and the resulting position is returned, else it continues.

Moving to the next step, the algorithm tries again to find a new position but this time by applying the provided items configuration policy. Based on the placement and the items configuration policy, the algorithm iterates again the placement area and tries to find a position. The placement area is iterated until the first available position is found or all the area are iterated. If an available position is found, then it is returned, otherwise it means that based on the provided parameters there is no available position in the given area and a new "bin" should be created. In the last step of the algorithm a new "bin" is created and the algorithm tries to find a position based on the given placement policy; if the given policy is meaningless, as in the case of and empty "bin" (e.g. user's position, or last opened position in the case that no specific data for the position of the user or the last opened position are provided), the extra configuration information is used. The algorithm gives priority first to the available on-surface's interaction position data, secondly to the position of the user in the room, thirdly to the last opened position of that item, and lastly the item is position in the center of the given area. In the end, all the available results are returned.

6.1.4 State manager

The state manager service is part of the CognitOS Classboard core services. The main task of this service is the Classboard's state management. In more detail, it controls all the state related services in order to prepare (on startup) and keep updated the current state of the Classboard. As it is depicted in Figure 45, the state related services are:

(i) CognitOS Classboard state
(ii) Boards state
(iii) Workspaces state
(iv) Windows state
(v) Applications state.

In the next sub sections, these services will be presented more analytically. In general, each one of them is related to a specific building block of the

Classboard and keeps information for it; the manager combines all of this information and build the current state of the Classboard. Both the state manager and the state related services interact and exchange information in a two-way communication. In more detail, the state manager handles all the requests coming from the Classboard's controller, e.g., 'Load active Board', 'Create a new workspace', 'Move a specific window', 'Launch an application', etc., and depending on the context of the requests it forwards them to the corresponding service to handle it. For example, the "load active Board" request is forwarded to the "CognitOS Classboard" service, the "create new workspace" is forwarded to the "workspace" service, the "open an application" to the "applications" service and so on. Finally, the state manager can restore the state of the Classboard, even if the Classboard has failed; the service keeps backups in the Datastore service to help the framework to recover from exceptional situations that cause the crush of the Classboard.

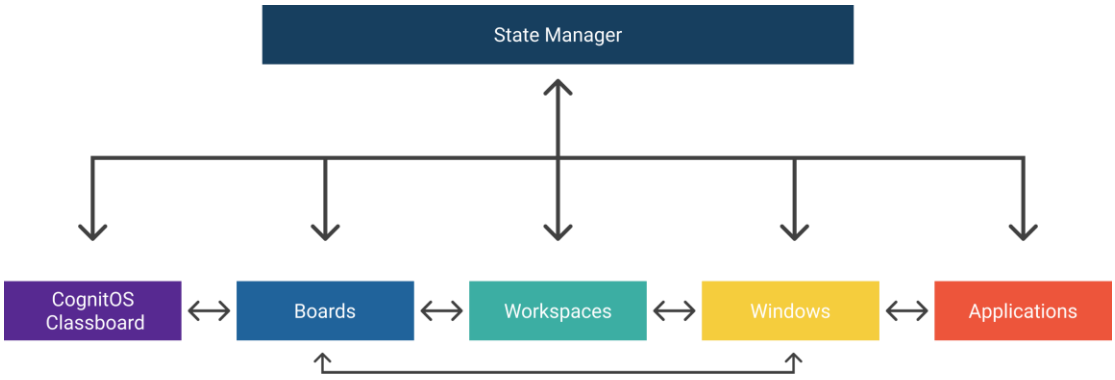


Figure 45: State related services and how they communicate.

Applications state

As it is implied by its name, the application service handles the state of the applications that have been installed in the Classboard. In more detail, it collects (e.g. updates coming from state manager or detected in the corresponding channel of the event federator) and keep stored information related to the state of that applications. When it is requested information for a specific application, the application state service searches in the Datastore in order to find the needed information. If the application is launched for first

time, no data will be found in the Datastore and that means that the application will be launched to its initial state. If the application had already been launched in the past, the service has stored information about the state of that application so as to collect and return them as an answer. As it will be described in next section in more detail, the service can keep the state of the applications directly or indirectly, depending on the application's configurations. In the first case, the service can keep directly information that describes the current state of the application and in the second one it keeps only an instance id which is provided by the application and will be used in order to specify to the applications server which state to load. In both cases, the information sent (either directly the state information or the instance id) to the application's server can reproduce the specific state in which the application was before it was closed. The service responds to requests such as:

- Bring me the list of all the state instances for the x application
- Bring me the state that the x application was the last time it launched
- Bring me a specific state (the state with id x)
- Update the state of x application.

Windows state

The windows state service is responsible for handling all the information that is related to the windows that hosts the applications. Each window corresponds to a specific instance of an application. The service collects information related to the state of the windows (e.g. information that coming from state manager, information that gathered from the event federator or the collected from the applications suite service), so as to keep always the state of the windows updated and safe by storing all the information in the Datastore. When it is requested information for a specific window, the service searches in the Datastore to retrieve the needed information. All this information constitutes the configuration data required to set up the window. In more detail, it manages information such as:

(i) the width and height of the window

(ii) dimension restrictions e.g. min, max width, height of the window

(iii) if the window can be resized or not

(iv) the current position of the window on the board

(v) if the window is pinned or not

(vi) if the border is enabled or disabled

(vii) if the window can be maximized

(viii) if the following functionality is enabled or not.

On the one hand, if the service does not find any information, this means that the application is launched for first time in the Classboard. In that case, the service requests the initial configuration information from the Applications suite service (e.g. dimension restrictions of the window) that is provided by the developers of the applications during the installation phase, and combines it with the default configurations of the Classboard (e.g. by default all the windows are resizable, movable and has borders). On the other hand, if the information is found in the Datastore the only action that is needed is to convert it in the appropriate form. As soon as the service collects all the information, the windows state service responds. In order to be more functional, sometimes the service requests directly information from the application state service, if it is needed (e.g. when it is requested to load a window, which means to provide as a response, beyond the information of the window, the information for the hosted application). In general, the service responds to requests such as:

- Is the window pinned?
- Is the window resizable?
- Which are the current dimension of the window?
- What are the dimension restrictions of the window?
- Load a specific window (both collect information about the window and the application state)
- etc.

Workspaces state

As it is implied by its name, this service handles the state of the created workspaces. The service is responsible for gathering information coming from the state manager service or published in the corresponding channels of the event federator so as to keep updated each workspace. In more detail, for each workspace it manages information such as:

(i) The list of alternatives groups of windows that constitutes that workspace

(ii) Which is it the visible group of windows

(iii) Which is the next group

(iv) Which is the previous group.

When it is requested to retrieve information for a specific workspace, the service searches in the Datastore and retrieves the requested information. In general, it handles requests such as:

- Create a new workspace
- Add or remove a specific workspace
- Add or remove a group of windows for a specific workspace
- Load next or previous group of windows.

As it has already been mentioned, each workspace consists of a number of alternative views, while each view contains a different set of Windows; the name and position of each window is stored in the workspace-specific state object. Thus, when the Classboard controller needs to restore or save a workspace, it communicates with the State Manager to handle the respective low-level actions (i.e. retrieve a past or save the current state in the Datastore). In general, it handles requests such as:

- Create a new empty view
- Add or remove a specific view
- Add or remove a window from a specific view
- Find the list of windows that exist in a specific view

- Load a whole view (prepare information about the list of the windows and furthermore extra information for each one of that windows).

Boards state

The Boards state service is responsible for handling all the information that is related to the state of the Boards. In more details, it collects information (e.g. coming from the state manager or from the event federator) and keeps it stored in the Datastore service. The main task of the service is to keep updated each Board by handling information such as the list of workspaces and the list of windows that are hosted in it. When it is requested information for a specific Board, the Boards state service searches and retrieves the corresponding information from the Datastore. Furthermore, when it is requested to load information regarding the entire Board, the service retrieves information for each workspace and window that exists in the requested Board; in this case, the service requests information from the workspaces and windows state services. In general, the service responds to requests such as:

- Create a new empty Board
- Add or remove a specific workspace
- Add or remove a specific window
- List the windows and workspaces that exists in a specific Board
- Load an entire Board.

CognitOS Classboard state

The CognitOS Classboard state service handles all the information that is related to the state of the system; this service provides an overview of the Classboard state. The CognitOS Classboard state manages requests coming from the state manager and events that are published in specific channels of the event federator; by merging this information it keeps up to date the state of the CognitOS Classboard. In more detail, it manages information such as:

(i) The list of Boards that constitutes the system

(ii) Which is the active Board

(iii) Which is the next Board

(iv) Which is the previous Board

When the state manager requests information from the service, it searches in the Datastore, it finds the needed information and then it responds. If the state manager requests to load completely all the information for the active Board, it interacts with the Boards state service to collect all the needed information. In more details, the service handles requests such as:

- Create a new Board
- Add or remove a specific Board
- Load next or previous Board.

6.1.5 Interaction with CognitOS Classboard

The CognitOS Classboard was designed featuring multimodal interaction methods, including multi-touch, touch gestures, mouse gestures, mid-air gestures, voice commands, and user position tracking. The interaction with the various elements of the CognitOS Classboard (i.e. Boards, Workspaces, Windows) can be divided in two categories: on-surface interaction and above-surface – or air-based – interaction.

On-surface Interaction service

In order to interpret user's on-surface interaction the "on-surface interaction" service was developed. Such service is responsible to handle the interaction related data generated from the board application. More specifically, when the user interacts with the system (using fingers, hands, mouse or interactive marker devices) a number of mouse, touch or pointer events occur. Those events are forwarded to the service to process them and recognize gesture patterns made by the user. As already mentioned, the service can recognize two different categories of gestures, namely "simple" and "advanced". A "simple" gesture is a way of combining pointing device or finger movements and clicks that the software recognizes as a specific event (e.g. tap, pan, press, etc.). On the other hand, an "advanced" gesture is a more complex gesture, in that case the software processes the input and tries to recognize shape,

number, and letter gesture patterns. In more detail, the service is split in two parts: (i) Simple gestures recognizer and (ii) Advanced gestures recognizer.

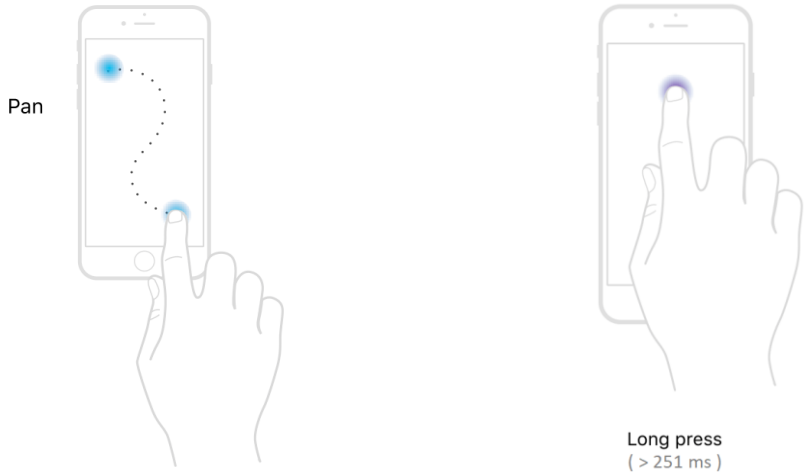
Simple gestures recognizer. As it is implied by its name, this module is responsible for simple gestures recognition. The recognizer is based on an external library called "hammer.js" [107] and extends its functionality. The library is available on node package manager [108]. Hammer.js is an open-source library that can recognize gestures made by touch, mouse, and pointer events; it is more convenient than other similar libraries because it does not have any dependencies and its size is small. Each time that new data are available, the simple gestures recognizer tries to find existing gestures. It combines the incoming events and if it recognizes a gesture, it emits the appropriate events using the event federator service (described in section 6.3.4). The service can recognize the following gestures:

a) The **Pan** gesture is recognized when the pointer, one or more fingers is down and moved around the screen (Figure 46a). When it is recognized, a pan event and some extra helpful events are emitted:
    - The **pan** event occurs when a pan gesture is detected.
    - The **pan-start** event occurs when the pan event is started.
    - The **pan-move** event occurs when the pointer is moving.
    - The **pan-end** event occurs when the pan event is completed.
    - The **pan-cancel** event occurs when an exception situation interrupts the pan gesture.
    - The **pan-left** event occurs when the pan gesture has direction to the left area of the screen.
    - The **pan-right** event occurs when the pan gesture has direction to the right area of the screen.
    - The **pan-up** event occurs when the pan gesture has direction to the top area of the screen.

- The **pan-down** event occurs when the pan gesture has direction to the bottom area of the screen.

b) The **Press** gesture is recognized when the pointer is down (one or more fingers, or a stylus touching the screen) at least for 251 milliseconds without any movement (Figure 46b). Two events are related to the press gesture:

- The **press** event occurs when the pointer is placed on the touch screen up to 251 milliseconds without any movement.
- The **press-up** event occurs when the pointer is removed from the touch screen.



(a) (b)

Figure 46: Describes in (a) a pan gesture [109], and in (b) a press gesture [110].

c) The **Rotate** gesture is recognized when two or more pointers are moving in a circular shape as depicted in Figure 47. When it is recognized, a rotate event and some extra helpful events are emitted:

- The **rotate** event occurs when a rotate gesture is detected.
- The **rotate-start** event occurs when the rotate gesture is started.
- The **rotate-move** event occurs when the pointers are moving.
- The **rotate-end** event occurs when the rotate gesture is completed.
- The **rotate-cancel** event occurs when the rotate gesture is interrupted.

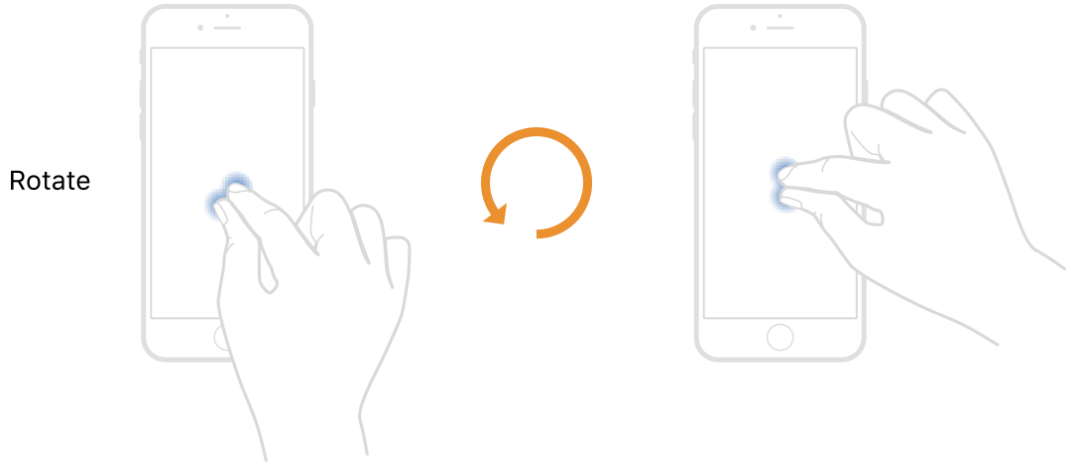


Figure 47: Describes a rotate gesture [111].

d) The **Pinch** gesture is recognized when two or more pointers (or two or more fingers) are moving toward (zoom-in) or away from each other (zoom-out) (Figure 48). More specifically, it is a continuous gesture that tracks the distance between the first two fingers that touch the screen. When it is recognized, a pinch event and some extra helpful events are emitted:

- The **pinch** event occurs when a pinch gesture is detected.
- The **pinch-start** event occurs when the pinch gesture is started.
- The **pinch-move** event occurs when the pointers are moving.
- The **pinch-end** event occurs when the pinch gesture is completed.
- The **pinch-cancel** event occurs when an exception situation interrupts the pinch gesture.
- The **pinch-in** event occurs when the pointers are moving toward each other.
- The **pinch-out** event occurs when the pointers are moving away from each other.

e) The **Swipe** gesture occurs when the pointer (or one or more fingers) is moving fast across the screen in a specific horizontal or vertical direction as depicted in Figure 49a. When it is recognized, a swipe event is emitted and some extra helpful events:

- The **swipe** event occurs when a swipe gesture is detected.
- The **swipe-left** event occurs when the swipe gesture has direction from the right to the left area of the screen.
- The **swipe-right** event occurs when the swipe gesture has direction from the left to the right area of the screen.
- The **swipe-up** event occurs when the swipe gesture has direction from the bottom of the screen to the top.
- The **swipe-down** event occurs when the swipe gesture has direction from the top to the bottom area of the screen.

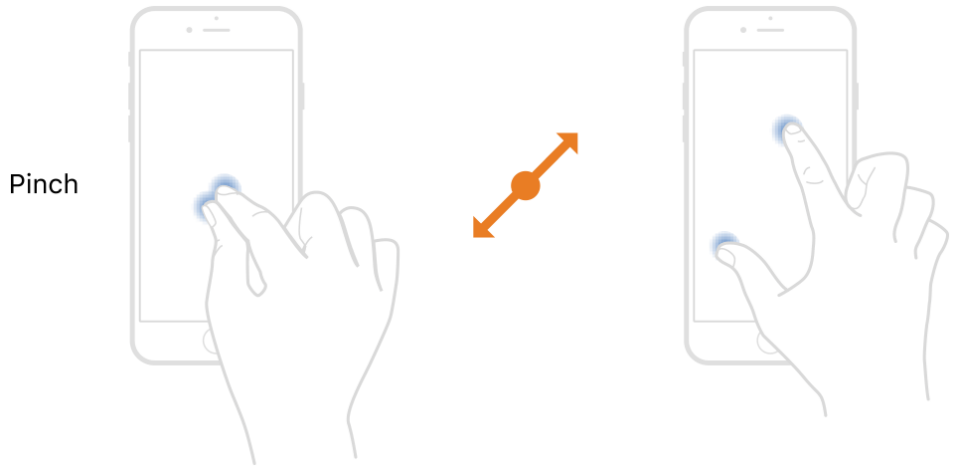


Figure 48: Describes a pinch gesture [112].

f) The **Tap** gesture is recognized when the pointer (or one or more fingers) is executing a small tap/click (Figure 49b). Additionally, multiple taps are recognized if they occur between an interval (i.e. maximum time 300 milliseconds), a maximum press time (i.e. about 250 milliseconds) and over a specific position. When the maximum interval time expires, a tap event is emitted containing as data the option tapCount (describes the number of taps occurred).

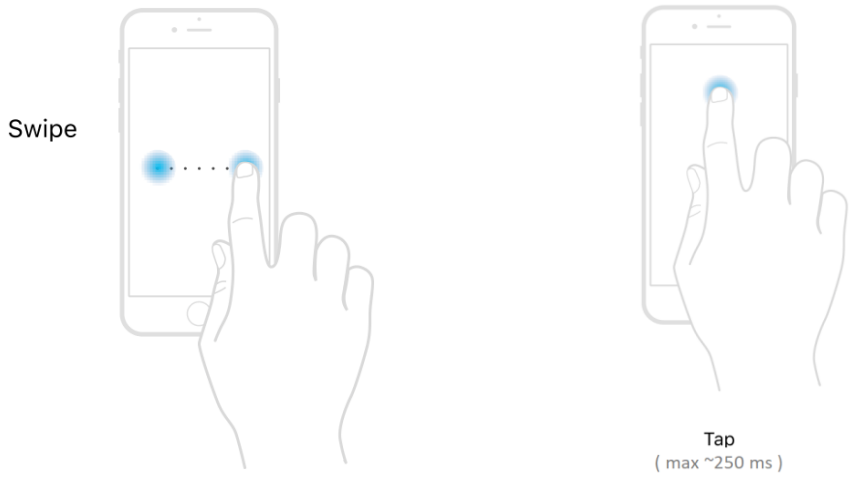


Figure 49: Describes in (a) swipe gesture [113], and in (b) tap gesture [114].

Advanced gestures recognizer

In order to recognize more complex gestures, the "advanced gestures recognizer" has been developed. This service is responsible to detect shape, number, and letter gesture patterns in the processing data. The recognizer extends the functionality of a software development kit called "MyScript interactive ink SDK" which is developed, maintained and provided by MyScript [115]. The MyScript SDK includes a powerful handwriting recognition engine and in order to be used it provides APIs for iOS, Android, Windows and web-based platforms. In the case of the advanced gestures recognizer, the API for web-based platforms is exploited and more specifically the version which uses web-sockets. In more detail, the service uses the engine's functionality so as to recognize only single characters, numbers or shapes. The recognition lifecycle is as follows:

Step 1: Initialize connection with MyScript's server and provide the context of the recognition

Information like the authorization data (i.e. application's id, application's key), the type of content that will be recognized (e.g. "handwritten text"), the language, etc.

Step 2: Retrieve and clean the generated data coming from the UI

The service receives the incoming data and converts them in the appropriate form. More specifically, it receives a series of mouse or touch points, it processes them and from each point it keeps only the x, y and timestamp attributes.

Step 3: Send the digested data to the server

The server receives the data, process them and starts to detect gestures.

Step 4: Detect gestures and recognition

The MyScript's recognition engine detects gestures in the given input and recognizes the content.

Step 5: Response

The server responds with the recognized information.

Step 6: Handle response

As soon as the "advanced gestures recognizer" receives the data, the response is processed and checked to assess if the recognized gesture is meaningful for CognitOS Classboard system. In more detail, a meaningful gesture is considered to be one that exists in the list of symbols and characters that can be handled via the system. For each meaningful action, the system emits the appropriate events using the event federator service (described in section 6.3.4).

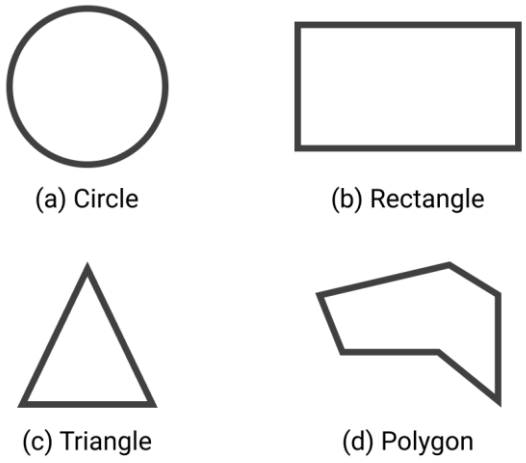


Figure 50: Visualizes the shapes that can be recognized.

The meaningful gestures which can be recognized are:

a) Letters of Latin alphabet
    - Lower case: a b c d e f g h i j k l m n o p q r s t u v w x y z
    - Upper case: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

b) Digits
    - 0 1 2 3 4 5 6 7 8 9

c) Symbols
    - Left arrow: ←
    - Up arrow: ↑
    - Right arrow: →
    - Down arrow: ↓
    - Less: <
    - Greater: >

d) Shapes
    - Circle (Figure 50a)
    - Rectangle (Figure 50b)
    - Triangle (Figure 50c)
    - Polygons (Figure 50d)

Above-surface Interaction service

In order to handle user's above-surface interaction the "above-surface interaction" service was developed. Such service is responsible to handle the data that are generated from the Classroom's environment services related to interaction (i.e. gestures recognition service, voice recognition service, position tracking service). More specifically, the environment of the classroom keeps track of each user inside the room (both teachers and students) and emits events depending on gestures, recognized voices commands, information about each user's position, about users' role (e.g. student, teacher), and other interaction related information. Those events are handled by the above-surface interaction service, they are filtered, enhanced

and emitted again using the event federator service (descripted in section 6.3.4). More specifically, the service filters out the useless events (i.e. events that does not make any sense for system's functionality) and keeps only the events that are exploited by the system; e.g. gestures or voice commands related to interaction with the board (e.g. swipe-up to navigate across boards, "create new board" to initialize a new board). Additionally, it keeps the position data that are related with the teacher and to the students if they are interacting with the Classboard.

Shortcuts library manager

The CognitOS Classboard framework provides a library of various shortcuts to the users. This library consists of a number of gestures, voice commands and other interaction related triggers which are paired to actions associated with the Classboard. In practice, each shortcut provides to the users an accelerated way of performing those actions. For the management of this library, the shortcuts library manger service was developed; as a matter of fact, this service is responsible to handle the existing shortcuts (it keeps all of the created pairs) and provides the opportunity to the users to create new ones. Therefore, the library manager is split into two parts a backend and a frontend.

Shortcuts library manager backend

This service is responsible to handle all the data that are related to the shortcuts of the CognitOS Classboard. It keeps stored both the predefined shortcuts, which is a set of shortcuts that had already created by the programmers of the framework, and another set of shortcuts that have been created by each user. The shortcuts are personalized, meaning that each user can define their own shortcuts. The set of predefined shortcuts is split into two different categories; the first one consists of shortcuts that are reserved by the system and cannot been updated by the users, and the second category which contains also shortcuts that are set by the system too, although the users can overwrite them and adjusts the action of these shortcuts to their preferences. A small sample of the defined shortcuts is described in Table 6. The system

allows the users to set different triggers to perform the same action. Furthermore, this service collects all the available triggers (i.e. gestures, voice commands, position) that are provided by the on-surface and above-surface interaction services; at the same time, it collects from the Classboard controller the actions that could be triggered. The list of those shortcuts (i.e. mappings between triggers and their respective actions) are used by this component to perform the right actions when such a case occurs (e.g. the user writes with the pen the letter "B" on an empty area).

Table 6: A sample of the defined shortcuts.

	Trigger	Action
On-surface	"B" gesture	Open Board manager
	"C" gesture	Switch of Classboard
	"M" gesture	Open menu
	"P" gesture	Open presentation viewer
	Circle shape	Capture the circled area
	"→" gesture	Go to next workspace
	"←" gesture	Go to previous workspace
	Swipe up (at the bottom of board)	Go to next board (works circular)
Above-surface	Clap hands twice	Go to next board
	Clap hands thrice	Go to previous board
	Swipe right hand	Go to next slide (applied on presentation viewer)
	Swipe left hand	Go to previous slide (applied on presentation viewer)
	"Start lesson" voice command	Load the content of the course
	"Pause lesson" voice command	Set the board in idle state
	"End lesson" voice command	Switch of Classboard

Shortcuts library manager frontend

The frontend part of the shortcuts library manager provides a user interface; via this interface the users (i) have an overview of the shortcuts that have

already been created, (ii) can update the existing shortcuts or (ii) create new ones. In more detail, the user has access to the list of all system- and user-created shortcuts. The user can distinguish the shortcuts that are reserved by the system from those that can be updated or can distinguish the system shortcuts from those that the user had previously created. More specifically, for each shortcut of the list information is provided such as what is the trigger, what is the action to be executed upon the trigger's occurrence, who has created this rule (e.g. the system or the user), if it is related to on-surface or above-surface interaction, if the trigger is a gesture or a voice command. As described above, the user can update the existing shortcuts (except the system reserved shortcuts). For example, the user can change the trigger or the action that will be executed and set a new trigger or another action. Finally, the user can create new shortcuts; the service follows the programming conditional statement "**if this, then that**" and creates chains of simple conditional statements or actually shortcuts. To create a new shortcut, the user in the first step should select the trigger of that shortcut (from the provided list of triggers) and then the action which will be executed when the trigger is enabled (from the provided list of actions). Both the lists of triggers and of actions are provided by the backend service of shortcuts library manager. After the creation of the new shortcut, the system is informed so as to handle it.

Interaction controller

The interaction controller is responsible to check if a shortcut is triggered or not, so as to inform the core controller; in order to perform this action, it gets access to the shortcuts library. The process consists of the steps below:

Step 1: Firstly, the interaction controller parses all the shortcuts kept in the shortcuts library; for each one, the controller subscribes, via the Event Federator (section 6.3.4) to the channel where the respective service publishes its events (i.e. the channel where the service that monitors the pen sends any recognized written shortcuts).

Step 2: The service waits for new events to occur. In practice, the service waits the Classboard or the classroom's environment to recognize gestures or voice commands.

Step 3: Each time a new gesture or voice command is recognized, an event is emitted and the interaction controller informs the core controller to execute the appropriate action. Those actions are described in the action part of that shortcut.
Step 4: The service continues waiting for new events to occur.

Finally, it is worth mentioning that the interaction controller is based on the messages mechanism which is provided by the event federator service and this mechanism gives the capability to the controller to handle simultaneously all of the shortcuts that are triggered.

6.1.6 Reachability recognition

The CognitOS Classboard is designed to be accessible to all people in the classroom regardless of their height. The board can be used simultaneously by the tall educator, the short young student and also by disabled users using wheelchairs. To address such requirement, the reachability recognition service has been developed. This service exploits information generated from the environment's services, such as the physical characteristics extraction, the user's recognition and the position tracking service. In more detail, the service obtains height related information from the user detection service, from the user's recognition information related to the role of the user or other helpful information that are stored in user's profile, information about the position of the user (e.g. how close he is to the board) from the homonymous service, and configuration information (i.e. the current distance between the board and the floor) from the Classboard configurator. When a user comes close to the board, the reachability recognition process starts; the service combines the extracted data and decides if the content of the board should be relocated or

not. More specifically, the service relocates only the window of the application that the user tries to use; that gives the opportunity to users with different height characteristics to interact simultaneously with different applications of the board which are adjusted to their characteristics. In the case of two users with different height who try to interact with the same application, the window is adjusted to the shorter one. As a result of the above, the content of the board is relocated in order to treat all users (e.g. younger students, disabled users using wheelchairs) in the same way and give them equal opportunities, no matter their height.

6.1.7 Classroom's bridge

The Classroom's bridge service is part of the core components of the Classboard; it is responsible to handle both all the outgoing requests to the classroom's environment and the incoming requests from the classroom's environment too.

The classroom's environment provides a number of useful services; that services are exploited by the CognitOS Classboard in order to enhance its functionality and to provide a better user experience (both to educators and students). The Classboard makes use of services laying in the categories of user's info, user's interaction, classroom state and utilities. Such services are accessible via the Classroom's bridge. In more detail, each one of the Classboard services that uses external services from the classroom should firstly make a request to the bridge; the bridge checks both the origin and the destination service and decides to forward the request or not. Furthermore, sometimes the service may have cached the response from previews requests and in order to speed up the process it returns to the requester immediately the response (when it is possible).

Additionally, the service handles the incoming requests from the Classroom. The environment needs to have access to the state of the board and other information (e.g. analytics); this helps the environment to set its state depending on the board's state or to keep statistics about the performance of

the framework, etc. The environment when it needs information from the Classboard firstly interacts with the bridge service. The bridge filters the incoming requests and decides based on some criteria to forward them or not, for example it checks if they are malicious, it checks if the Classboard is overloaded by tasks, etc.

Finally, the bridge helps the framework to improve its performance and keep a balance between the outgoing and the incoming requests.

6.1.8 CognitOS Classboard as a service

The CognitOS Classboard framework is built based on a micro-service architecture style. It is designed and developed in order to interoperate with other systems and services of the Classroom's environment described in Chapter 4 (e.g. CognitOS, Lector, Classmate, AmI-Solertis) and artifacts (e.g. intelligent student desk, educator's workstation). For example, it retrieves data offered by the environment's systems and services (such as profiling data provided by the Classmate), and at the same time it generates and outputs data to the environment, which other services can consume on demand (e.g. sent a screen capture to the intelligent student desks).

The CognitOS Classboard offers an API enabling the environment to perform various actions and manage the overall state of the framework. Those functions are grouped in five categories, which control: (i) the system (start, pause, stop), (ii) the board, (iii) the workspaces, (iv) the windows and (v) the applications. For example, the CognitOS system can launch specific applications on the board when LECTOR detected a problematic situation, or the educator -using his tablet-can remotely control the Classboard (e.g. launch an application, make an annotation, rearrange the contents of the board). Currently, the API consists of the functions listed in Table 7; however, new functions are added frequently, since the CognitOS Classboard is an ever-growing system.

Table 7: CognitOS Classboard API functions.

Category	Functions
System	switchOn
	switchOff
	idle
Board	getListOfBoards
	getInfo
	create
	loadNext
	loadPrevious
	load
	launchApplication
	capture
	loadAnnotation
Workspace	getInfo
	createNew
	createAlternativeView
	loadNext
	loadPrevious
	load
	launchApllication
	capture
	loadAnnotation
	close
	move
	resize
Window	getInfo
	close
	maximize
	restoreDown

	pin
	followMe
	frame
	loadAnnotation
	capture
	move
	resize
	shareContent
Apps	getInfo
	getListOfApplications
	getExtraFunctionality

System

It is provided a number of functions that are applied to the system:

1 Switch on

Switches on the CognitOS Classboard

Resource URL	
POST https:// classboard.ics.forth.gr/api/system/switchOn	
Resource Information	
Response format	JSON
Parameters	
(none)	
Example Response	

```
{
     “success”: true
}
```

2 Switch off

Switches off the CognitOS Classboard

Resource URL	
POST https://classboard.ics.forth.gr/api/system/switchOff	
Resource Information	
Response format	JSON

Parameters

(none)

Example Response

```
{
      “success”: true
}
```

3 Idle

Set the CognitOS Classboard in idle state

Resource URL

POST https://classboard.ics.forth.gr/api/system/idle

Resource Information

Response format	JSON

Parameters

(none)

Example Response

```
{
      “success”: true
}
```

Board

It is provided a number of functions that are applied to the Boards:

1 Get list of Boards

Get the list of the created Boards

Resource URL

POST https://classboard.ics.forth.gr/api/board/getListOfBoards

Resource Information

Response format	JSON

Parameters

(none)

Example Response

```
{
      “boards”: [“board_1”, “board_2”, “board_3”]
}
```

2 Get info

Get information for a specific Board (e.g. the windows and the workspaces that resides on that Board)

Resource URL

`POST` https://classboard.ics.forth.gr/api/board/getInfo

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
board_id	true	Contains the requested board's id	“board_1”

Example Request

```
{
	“board_id”: “board_1”
}
```

Example Response

```
{
	“board_id”: “board_1”,
	“windows”: [“window_1”, “window_2”, “window_3”],
	“workspaces”: [“workspace_1”, “workspace_2”],
	“success”: true
}
```

3 Create

Create a new Board

Resource URL

`POST` https://classboard.ics.forth.gr/api/board/create

Resource Information

Response format	JSON

Parameters

(none)

Example Response

```
{
	“board_id”: “board_4”,
	“success”: true
}
```

4 Load next

Load the next alternative Board

Resource URL

`POST https://classboard.ics.forth.gr/api/board/loadNext`

Resource Information

Response format	JSON

Parameters

(none)

Example Response

```
{
     “board_id”: “board_2”,
     “success”: true
}
```

5 Load previous

Load the previous alternative Board

Resource URL

`POST https://classboard.ics.forth.gr/api/board/loadPrevious`

Resource Information

Response format	JSON

Parameters

(none)

Example Response

```
{
     “board_id”: “board_1”,
     “success”: true
}
```

6 Load

Load a specific board

Resource URL

`POST https://classboard.ics.forth.gr/api/board/load`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
board_id	true	Contains the requested board's id	“board_3”

Example Request

```
{
    “board_id”: “board_3”
}
```

Example Response

```
{
    “board_id”: “board_3”,
    “success”: true
}
```

7 Launch application

Launch an application on the Board

Resource URL

POST https://classboard.ics.forth.gr/api/board/launchApplication

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
application _id	true	Contains the requested application's id	“application _3”

Example Request

```
{
    “application_id”: “application_3”
}
```

Example Response

```
{
    “application_id”: “application_3”,
    “success”: true
}
```

8 Capture

Capture the contents of the whole Board

Resource URL

`POST` https://classboard.ics.forth.gr/api/board/capture

Resource Information

Response format	JSON

Parameters

(none)

Example Response

```
{
      “capture_url”: “https://classboard.ics.forth.gr/api/storage/capture_1”,
      “success”: true
}
```

9 Load annotation

Load on the Board an annotation that was drawn remotely

Resource URL

`POST` https://classboard.ics.forth.gr/api/board/loadAnnotation

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
points	true	Contains the x,y points that will be drawn on the board	`[(0,0),(0,1), ...]`

Example Request

```
{
      “points”: [(0,0), (0,1), (0,2), (0,3)]
}
```

Example Response

```
{
      “success”: true
}
```

Workspace

It is provided a number of functions that are applied to the Workspaces:

1 Get info

Get information for a specific Workspace (e.g. the windows that resides in that Workspace)

Resource URL

`POST https://classboard.ics.forth.gr/api/workspace/getInfo`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the requested workspace's id	`"workspace_1"`

Example Request

```
{
     "workspace_id": "workspace_1"
}
```

Example Response

```
{
     "workspace_id": "workspace_1",
     "windows": ["window_1", "window_2", "window_3"],
     "alternatives": ["workspace_1.1", "workspace_1.2"],
     "configurations": {
          position: {
               x: 0,
               y: 5
          },
          dimensions: {
               rows: 5,
               columns: 10
          }
     }
     "success": true
}
```

2 Create new

Create a new Workspace

Resource URL

POST https://classboard.ics.forth.gr/api/workspace/createNew

Resource Information

Response format	JSON

Parameters

(none)

Example Response

```
{
      “workspace_id”: “workspace_4”,
      “success”: true
}
```

3 Create alternative view

Create an alternative view for a specific Workspace

Resource URL

POST
https://classboard.ics.forth.gr/api/workspace/createAlternativeView

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace’s id that will be extended with an extra alternative view	“workspace_1”

Example Request

```
{
      “workspace_id”: “workspace_1”
}
```

Example Response

```
{
      “workspace_id”: “workspace_1”,
      “alternative_workspace_id”: “alternative_workspace_2”
      “success”: true
}
```

4 Load next

Load the next alternative view of a Workspace

Resource URL

`POST https://classboard.ics.forth.gr/api/workspace/loadNext`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace's id that will be updated	"workspace_3"

Example Request

```
{
    "workspace_id": "workspace_3"
}
```

Example Response

```
{
    "workspace_id": "workspace_3",
    "alternative_workspace_id": "alternative_workspace_3.3"
    "success": true
}
```

5 Load previous

Load the previous alternative view of a Workspace

Resource URL

`POST https://classboard.ics.forth.gr/api/workspace/loadPrevious`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace's id that will be updated	"workspace_3"

Example Request

```
{
    "workspace_id": "workspace_3"
}
```

Example Response

```
{
      “workspace_id”: “workspace_3”,
      “alternative_workspace_id”: “alternative_workspace_3.2”
      “success”: true
}
```

6 Load

Load a specific alternative view of a Workspace

Resource URL

POST https://classboard.ics.forth.gr/api/workspace/load

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace’s id that will be updated	“workspace_3”
alternative_ workspace_id	true	Contains the requested alternative workspace id	“alternative_ workspace_3.2”

Example Request

```
{
      “workspace_id”: “workspace_3”,
      “alternative_workspace_id”: “alternative_workspace_3.2”
}
```

Example Response

```
{
      “workspace_id”: “workspace_3”,
      “alternative_workspace_id”: “alternative_workspace_3.2”
      “success”: true
}
```

7 Launch application

Launch an application on a Workspace

Resource URL

`POST`
`https://classboard.ics.forth.gr/api/workspace/launchApplication`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace's id that will be updated	`"workspace_3"`
application _id	true	Contains the requested application's id	`"application_1"`

Example Request

```
{
      "workspace_id": "workspace_3",
      "application_id": "application_1"
}
```

Example Response

```
{
      "workspace_id": "workspace_3",
      "application_id": "application_1",
      "success": true
}
```

8 Capture

Capture the contents of a Workspace

Resource URL

`POST` `https://classboard.ics.forth.gr/api/workspace/capture`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace's id that will be captured	`"workspace_3"`

Example Request

```
{
      “workspace_id”: “workspace_3”
}
```

Example Response

```
{
      “workspace_id”: “workspace_3”
      “capture_url”: “https://classboard.ics.forth.gr/api/storage/capture_1”,
      “success”: true
}
```

9 Load annotation

Load on a Workspace an annotation that was drawn remotely

Resource URL

POST
https://classboard.ics.forth.gr/api/workspace/loadAnnotation

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace’s id that will be updated	“workspace_3”
points	true	Contains the x,y points that will be drawn on the board	[(0,0),(0,1), ...]

Example Request

```
{
      “workspace_id”: “workspace_3”,
      “points”: [(0,0), (0,1), (0,2), (0,3)]
}
```

Example Response

```
{
      “workspace_id”: “workspace_3”,
      “success”: true
}
```

10 Close

Close a Workspace (the workspace should be empty of windows)

Resource URL

`POST https://classboard.ics.forth.gr/api/workspace/close`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace's id that will be closed	"workspace_3"

Example Request

```
{
      "workspace_id": "workspace_3"
}
```

Example Response

```
{
      "workspace_id": "workspace_3",
      "success": true
}
```

11 Move

Change the position of a Workspace

Resource URL

`POST https://classboard.ics.forth.gr/api/workspace/move`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace's id that will be moved	"workspace_3"
position	true	Contains the new position of the workspace on the board	{x:0, y:5}

Example Request

```
{
      "workspace_id": "workspace_3",
```

```
      “position”: {
            x: 0,
            y: 5
      }
}
```

Example Response

```
{
      “workspace_id”: “workspace_3”,
      “success”: true
}
```

12 Resize

Changes the dimensions of a Workspace

Resource URL

POST https://classboard.ics.forth.gr/api/workspace/resize

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
workspace_id	true	Contains the workspace’s id that will be resized	“workspace_3”
dimensions	true	Contains the new dimensions of the workspace	{rows:5, columns:10}

Example Request

```
{
      “workspace_id”: “workspace_3”,
      “position”: {
            x: 0,
            y: 5
      }
}
```

Example Response

```
{
      “workspace_id”: “workspace_3”,
      “success”: true
}
```

Window

It is provided a number of functions that are applied to the Windows:

1 Get info

Get information for a Window (e.g. dimensions, hosted application, etc.)

Resource URL

`POST https://classboard.ics.forth.gr/api/window/getInfo`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the requested window's id	"window_1"

Example Request

```
{
      "window_id": "window_1"
}
```

Example Response

```
{
      "window_id": "window_1",
      "hosted_application": "application_1"
      "configurations": {
            position: {
                  x: 0,
                  y: 5
            },
            dimensions: {
                  rows: 5,
                  columns: 10
            }
      }
      "success": true
}
```

2 Close
Close a Window

Resource URL

`POST` https://classboard.ics.forth.gr/api/window/close

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window's id that will be closed	"window_1"

Example Request

```
{
	"window_id": "window_1"
}
```

Example Response

```
{
	"window_id": "window_1",
	"success": true
}
```

3 Maximize
Maximizes a Window

Resource URL

`POST` https://classboard.ics.forth.gr/api/window/maximize

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window's id that will be maximized	"window_1"

Example Request

```
{
	"window_id": "window_1"
}
```

Example Response

```
{
	"window_id": "window_1",
	"success": true
}
```

4 Restore down

Restore a window to its initial state (before maximized)

Resource URL

`POST https://classboard.ics.forth.gr/api/window/restoreDown`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window's id that will be restored down	`"window_1"`

Example Request

```
{
    "window_id": "window_1"
}
```

Example Response

```
{
    "window_id": "window_1",
    "success": true
}
```

5 Pin

Pin a Window to a specific position

Resource URL

`POST https://classboard.ics.forth.gr/api/window/pin`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window's id that will be pinned	`"window_1"`
pin	true	Contains information about the pin state of the window	`true`

Example Request

```
{
      “window_id”: “window_1”,
      “pin”: true
}
```

Example Response

```
{
      “window_id”: “window_1”,
      “success”: true
}
```

6 Follow-me

Enables of disables follow me functionality for a Window

Resource URL

POST https://classboard.ics.forth.gr/api/window/followMe

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window’s id that will be affected	“window_1”
follow_me	true	Contains information about the follow_me state of the window	true

Example Request

```
{
      “window_id”: “window_1”,
      “follow_me”: true
}
```

Example Response

```
{
      “window_id”: “window_1”,
      “success”: true
}
```

7 Frame

Shows or hides the frame of a Window

Resource URL

`POST https://classboard.ics.forth.gr/api/window/frame`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window's id that will be affected	"window_1"
frame	true	Contains information about the window's frame	true

Example Request

```
{
      "window_id": "window_1",
      "frame": true
}
```

Example Response

```
{
      "window_id": "window_1",
      "success": true
}
```

8 Load annotation

Load on a Workspace an annotation that was drawn remotely

Resource URL

`POST https://classboard.ics.forth.gr/api/window/loadAnnotation`

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window's id that will be affected	"window_1"
points	true	Contains the x,y points that will be drawn on the board	[(0,0),(0,1),...]

Example Request

```
{
      “window_id”: “window_1”,
      “points”: [(0,0), (0,1), (0,2), (0,3)]
}
```

Example Response

```
{
      “window_id”: “window_1”,
      “success”: true
}
```

9 Capture

Capture the contents of a Window

Resource URL

POST https://classboard.ics.forth.gr/api/window/capture

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window’s id that will be affected	“window_1”

Example Request

```
{
      “window_id”: “window_1”
}
```

Example Response

```
{
      “window_id”: “window_1”
      “capture_url”: “https://classboard.ics.forth.gr/api/storage/capture_1”,
      “success”: true
}
```

10 Move

Change the position of a Window

Resource URL

`POST` https://classboard.ics.forth.gr/api/window/move

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window's id that will be moved	"window_1"
position	true	Contains the new position of the window	{x:0, y:5}

Example Request

```
{
      "window_id": "window_1",
      "position": {
            x: 0,
            y: 5
      }
}
```

Example Response

```
{
      "window_id": "window_1",
      "success": true
}
```

11 Resize

Changes the dimensions of a Window

Resource URL

`POST` https://classboard.ics.forth.gr/api/window/resize

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window's id that will be resized	"window_1"

dimensions	true	Contains the new dimension of the window	`{rows:5, columns:10}`

Example Request

```
{
      “window_id”: “window_1”,
      “dimensions”: {
            rows: 5,
            columns: 10
      }
}
```

Example Response

```
{
      “window_id”: “window_3”,
      “success”: true
}
```

12 Share content

Get a link to the content of an application

Resource URL

POST https://classboard.ics.forth.gr/api/window/shareContent

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
window_id	true	Contains the window’s id that will be requested its content	“window_1”

Example Request

```
{
      “window_id”: “window_1”
}
```

Example Response

```
{
      “window_id”: “window_1”,
      “content”: “https://classboard.ics.forth.gr/api/storage/capture_1”,
      “success”: true
}
```

Applications

It is provided a number of functions that are applied to the Applications:

1 Get list of applications

Get the list of the applications that compose CognitOS Classboard applications library

Resource URL

POST
https://classboard.ics.forth.gr/api/application/getListOfApplications

Resource Information

Response format	JSON

Parameters

(none)

Example Response

```
{
      “applications”: [“application_1”, “application_2”]
}
```

2 Get info

Get information for a specific application

Resource URL

POST
https://classboard.ics.forth.gr/api/application/getApplication

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
application_id	true	Contains the requested application’s id	“application_1”

Example Request

```
{
      “application_id”: “application_1”
}
```

Example Response

```
{
      “application_id”: “application_1”,
      “url”: “http://www.google.com/”
      “success”: true
}
```

3 Get extra functionality

Get a list of the extra actions that exposed via a specific application

Resource URL

POST
https://classboard.ics.forth.gr/api/window/getExtraFunctionality

Resource Information

Request format	JSON
Response format	JSON

Parameters

Name	Required	Description	Example
application_id	true	Contains the requested application's id	"application_1"

Example Request

```
{
      "application_id": "application_1"
}
```

Example Response

```
{
      "application_id": "application_1",
      "functionality": [
            {
                  name: "undo",
                  url: "https://www.annotation.gr/action/undo"
            },
            {
                  name: "redo",
                  url: "https://www.annotation.gr/action/redo"
            },
            {
                  name: "clean",
                  url:"https://www.annotation.gr/action/clean"
            }
      ]
      "success": true
}
```

6.2 User Interface Framework

The user interfaces framework groups all the services that are related to the applications that are used to set up the Classboard's state before the start of the lecture, the application that visualize the content of the Classboard, and two extra applications that are used to control the Classboard's content during the lecture. In more detail, a Desktop, a Classboard, a Tablet, and a Mobile application are provided. Furthermore, a utility service is provided too, which is responsible to provide configuration and content related information to the user interface applications. From a programming perspective, those interfaces are build using angular [116] and exploits packages that are provided by the Node Package Manager (NPM) [108].

6.2.1 Implementation of fundamental UI components

In order to manage the content of the Classboard five major components have been created (i.e. board, workspace, window, application). Each one of them is represented as a different graphical component in the Classboard application.

- The **CognitOS Classboard** component is the major component of the CognitOS Classboard. This component constitutes the basis of the user interface, on which other components are placed; it stores the list of all the created boards and information about their order.
- The **board** component is a grid area on which workspaces and windows are placed; in practice, each board is split in a number of rows and columns. This grid helps to manage the content of the board in an easier manner (e.g. position items in specific positions, move them, etc.). It exploits the functionality of an external library called "angular-gridster2" [117], [118], which provides mechanisms to create and add new items, resize them, move them, etc. Additionally, it handles the physics during the movement of the elements such as the conflicts when one item pushes another. Finally, this library satisfies one of the main principles of the CognitOS Classboard which is "the items of the board must not overlap".

- The **workspace** component is a grid area which is split in rows and columns. This component is built based on the external library "angular-gridster2" [117], [118]. That area can host only window components. As already described, each workspace consists of a number of alternative views (i.e. groups of hidden workspaces), while only one view can be visible at any given time; the workspace holds the necessary information about the existing alternative views such as their order, the windows that belongs to each one, any dimensions' limitations of a workspace, etc.
- The **window** component is the container that act as a host of an application component. Each window keeps configuration information related to its state such as its position, its current dimensions (e.g. if the window is maximized) and its dimensions' limitations (max, min width / height) that govern its resizing policy (e.g. allow or restrict resizing). At the same time, it keeps information about its pinned state (e.g. if it is permitted to be moved or not) and the visibility of the window's frame. Finally, when it closes, it informs the hosted application in order to save its current state and restore to it the next time that will be launched.
- The **application** component presents the User Interface of the launched application. During the resizing of its parent window, the application (and consequently its content) automatically adjusts to the new dimensions.

6.2.2 Decomposition of Classboard's UI layers

The CognitOS Classboard is composed of a number of different layers. Each one of those layers is responsible for different tasks and hosts different items. More specifically, the application is consisted of five layers (i) board, (ii) follow-me, (iii) annotation, (iv) capture and (v) gestures. The layers are ordered as depicted in Figure 51; at the bottom of the hierarchy there is the board layer, then the follow-me layer, then the annotation, the capture and on the top of hierarchy the gestures layer.

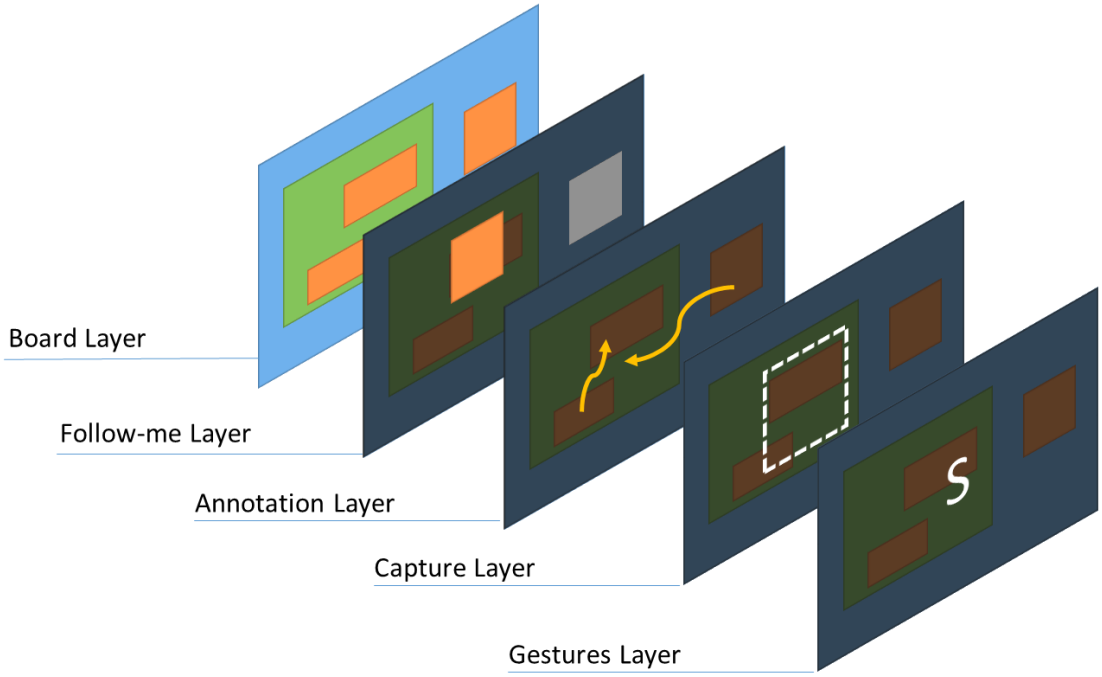


Figure 51: Classboard's user interface layers.

Each layer has a different role in the application:

- The **board** layer hosts the main content of the application (e.g. workspaces, windows, menu, etc.).
- The **follow-me** layer hosts the windows that are selected by the user to follow him.
- The **annotation** layer is used by the annotation tools in order to visualize the annotations that are made by the user.
- The **capture** layer is used by the capture tool; the user can select and take captures of specific areas of the display.
- The **gestures** layer is related to the on-surface gestures made by the user.

In order to accomplish all the needed tasks, each layer makes use of the events (e.g. mouse, touch events) that are generated during the user's interaction with the application. In practice, the application exploits the JavaScript event propagation mechanism.

JavaScript Event Propagation. As it is explained in [119], "*the event propagation is a mechanism that defines how events propagate or travel the DOM tree to arrive at its target and what happened to that target afterward*". Each time that an event is generated, it travels up and down through the DOM tree to reach its target. The event propagation mechanism has three different phases:

a) **Capturing phase** – the event goes down to the element. In more detail, in the capturing phase the events propagate from the window down through the DOM tree to the target element. During the travel to the target element if any ancestor (i.e. parent, grandparent, etc.) of the target element and the target itself a registered capturing event listener for that type of event, those listeners are executed during this phase.
b) **Target phase** – the event reached the target element. This phase occurs when the event arrives at the target element that has generated the event. If the element has registered handlers for that specific type of event, they are executed.
c) **Bubbling phase** – the event bubbles up from the element. In more detail, in the bubbling phase the events propagate from the target element back the window. During the travel to the window, the event visits all of the ancestors of the target element. Also, if any ancestor of the target element has event handlers registered for that type of event, those handlers are executed during this phase.

Additionally, an important function of the event propagation mechanism is that the propagation of the event can be stopped. In more details, if a registered handler catches an event, it can stop the propagation of that event and prevent any other elements' event handler from being notified about the event.

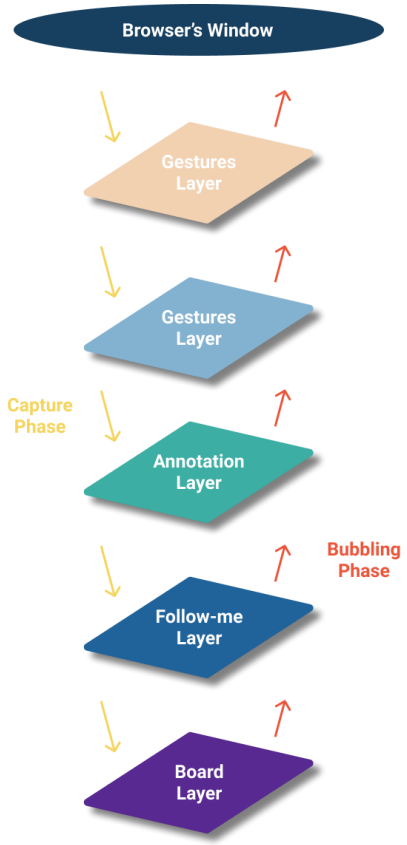


Figure 52: Events travel from window to target element and back to window.

CognitOS Classboard and Events handling. The CognitOS Classboard makes use of the event propagation mechanism in order to accomplish all the needed tasks and enhance its functionality. Each one of the generated events travels from the DOM's window element to the target element end returns back to the window. During this travel, each event passes from the intermediate layers of the application. As depicted in Figure 52 (supposing that the target element is hosted in the board layer) an event starts from the window, during the capturing phase it goes down to the target element and passes though the gestures, the capture, the annotation, the follow-me layers and arrives to the element on the board layer (target phase). Then during the bubbling phase, the event returns back to the DOM's window element passing though the same elements as in the capturing phase, but in reverse order (follow-me layer, annotation layer, capture, gestures layer). Each one of those layers has registered event handlers for different types of events; in some cases,

depending on the type of handling event and the state of the layer the propagation to other layers are stopped. More analytically:

- The **gestures layer** is responsible to handle all the events (i.e. mouse, touch events) that are passing through that layer (during the capturing phase). All the caught events are provided to the on-surface interaction service. The interaction service combines the generated events and recognizes the gestures made by the user. When a new gesture is recognized the system interprets it and takes the appropriate actions. The events related to gesture do not propagated to the next layers.
- The **capture layer** is responsible for handling all the events (both mouse and touch events) during the capturing phase only if the capture mode is on. In general, this layer is deactivated and the events pass through it. When the user enables the capture tool, the layer is activated and the user can select a specific area of the board and capture it.
- The **annotation layer** is responsible to handle all the mouse and touch events (during the capturing phase) only if the annotation mode is on. In general, this layer is deactivated; that means that it exists in the DOM but does not handle the events that pass through it. When the user enables the annotation, the layer is activated and starts handling all the events. Each event that is handled does not propagated to the next layers.
- The **follow-me layer** is responsible for handling the mouse and touch events that are related to the specific layer (during the capturing phase). Similar to the annotation layer, the follow-me layer in general is deactivated. It is activated when the follow-me functionality is enabled; in that case the layer handles all the events that triggers the event handlers of that layer and prevent them to travel to the next layers.
- The **board layer** is at the bottom of the layers' hierarchy; this layer exploits the generated events in order to update the state of the active board (e.g. move items, interact with the content of the launched applications, etc.). When an event is handled by a specific application or the board, it is not propagated back to the DOM's window.

6.3 CognitOS Classboard Utilities

CognitOS Classboard also offers a set of services which are designed to help configure and maintain the framework, keep stored the generated data of the services, provide asynchronous communications between the framework's services and a connection with the services that are provided by the AmI environment. In more detail, those services (i.e. state monitoring) are aimed to support developers in maintaining the framework by collecting information about the state of the exploiting services and providing an overview of the framework's state. Furthermore, to configure the state of the framework, a board configurator and remote controllers' manager are provided. The Datastore service is used to handle the generated document and binary data. Finally, a number of meta-services are developed in order to handle the data coming from the AmI environment.

6.3.1 State monitoring

As already mentioned, the CognitOS Classboard is based on the micro service architecture. The framework is split into specialized containers designed to perform a specific task or process, and micro services enable each component to operate independently. In comparison to monolithic applications, the micro services approach on the one hand has plenty of advantages such as grater agility, better scalability, faster development, etc., but on the other hand it has a number of disadvantages too, such as global testing is difficult, debugging problems can be harder, it is more difficult to monitor because of the complexity of the architecture, etc. Furthermore, focusing on CognitOS Classboard design and more specifically on the Applications Suite which consist of a large number of external applications, the main concern is the availability of those services, so as to keep the Classboard functional even if the external applications fail. As a result of those disadvantages and concerns, the state monitoring service has been developed. The provided information

(i.e. service availability and the health state of the services) is used by CognitOS Classboard in order to avoid failures (e.g. if an external application is offline it tries to find and propose similar applications from the Applications Suite) and for the programmers to have an overview of the state of each service (e.g. via the provided interface, when the service discovers a failure it send alert messages, etc.).

State monitoring backend

The state monitoring service is responsible for collecting and providing real-time data related to the state both of the internal micro services of CognitOS Classboard and the external services of the Classroom's environment and the external applications too. More specifically, the main task of the service is to check if all those services are available (are online or offline) and they are running correctly. In the first case, the state monitoring service checks the availability of the services by sending periodically requests to that services; if a response is returned, this means that the service is online, otherwise it is offline. Additionally, the service checks if the online services are running correctly too; to achieve that extra functionality, the services use their own test checking mechanism (as described in section 6.4.1). The state monitoring service makes requests to the provided URLs and gets in the response as an answer if the services are running correctly or not. As a result, each service's state (i.e. internal or external) could characterized as "on-line", "off-line", or "pay-attention". In the first case, an "on-line" the service is available and fully functional; in the second one, an "off-line" service could not be reached (e.g. the server where the application is hosted has problem); and in the last case, "pay-attention", the service is available but the checking test has failed, that means it is not working correctly and probably if it used it may cause problems to the system. All that information is provided to the CognitOS Classboard internal services and to the developers of the framework too.

6.3.2 Remote controllers' manager

This service manages the interaction data that are generated via the remote controllers. It is responsible for handling all the data that are generated by the desktop, the tablet and the mobile applications and are related to updates that should be performed on the board at run time. The controller filters the collected data and based on the state of the board decides if those updates should affect the board or not.

6.3.3 Board Configurator service

During the initialization phase of the Board's applications, a bunch of configuration data are needed to set-up the state of each application (e.g. number of boards, layout of specific workspaces, active windows during a past session). In order to collect those data and provide them to the applications the configurator service has been developed.

6.4 Classroom services library

The classroom's environment is technologically enhanced; it is equipped with various ambient facilities (e.g. projectors, Kinect devices, cameras, smart sensors, motorized blinds, smart lights, etc.). In order to control all of those devices and sensors, a number of micro services are available in the ICS-FORTH smart classroom environment. These services are available to researchers and programmers developing smart classroom applications. In the case of CognitOS Classboard, a large number of the existing services were used, belonging to the categories: (i) environment's state, (ii) users' interaction, (iii) users' info. A number of meta-services are developed (that act as bridges to the classroom's environment services functionality) in order to handle the data that generated by the Classroom's environment services. All these bridges services are grouped in the Classroom's services library.

Classroom controller

The "classroom controller" service has been developed In order to manage the physical environment. This service is responsible for managing all the services of the environment and enhance their functionality. Additionally, it handles all incoming requests (handles request and allows or blocks their execution).

The main goal of the service is to support the environment's services and at the same time the classroom's environment. The service aims to help the environment to be sensitive, adaptive and responsive to human presence. In more details, the classroom's controller supports the environment in interpreting the actions of the users, managing the state of the room and adjusting it based on contextual information; for example, when a presentation starts, it switches off the room's lights or when is time for a break it opens the curtains and the blinds, etc. Simultaneously, the controller monitors the incoming requests related to the services and applications of the board and permits or blocks their execution (e.g. block multiple simultaneous requests from the same service or unauthorized requests).

6.4.1 Environment's state

This category contains all the services that perform actions related to the state of the classroom's environment. The role of those services is to handle the state of the classroom, by updating directly the state of the classroom or providing information about the environment (e.g. the temperature of the room). More specifically, the following services are used:

Temperature

This service is responsible for keeping track of the room's temperature and providing data such as the current or past values of the room's temperature.

Lights

The Lights service controls the state of lights (e.g. the lamps, the light strips, etc). There are many properties that can be controlled using this service, for example:

a. Turn on or off the lights.
b. Adjust the brightness.
c. Change the color of the lights.
d. Update the hue of the lights.
e. Change the saturation of the color.

Windows blinds

The Windows blinds service manages the blinds of the windows. This service provides controls to open and close the blinds, get information about their state, set daily and weekly schedules, etc.

Windows curtains

The Windows curtains service is responsible for opening or drawing the curtains and providing information about the state of the curtains in the room.

State Controller

In order to behave intelligently, the CognitOS Classboard install in the classroom 's environment, via AmI-Solertis (section 4.1.1), the "State Controller" service. This service aims to enhance the overall user experience by synchronizing any board-related actions with the necessary actions in the intelligent space (e.g. appropriately adjust the windows blinds and room's ambient lights when a full-screen video starts in a sunny day).

6.4.2 Interaction Manager

This group consists of various services related to the interaction in the ambient intelligence classroom's environment. Those services are targeted to enabling natural, intuitive, high-quality, unobtrusive, inclusive and fault-tolerant

interaction of the users (i.e. students, teachers) with the classroom's environment via multiple modalities and devices, such as recognition and monitoring of users' interaction with the environment, and multimodal interaction techniques. The software bridges described below were created in the context of CognitOS Classboard system to exploit existing services:

Position tracking Bridge

The classroom's environment is always tracking the positions of all the people inside the room (both teachers and students). This service at any time provides information about the positions of each person in the class and some extra info such as their body position (e.g. sitting, moving, standing, raising the hand). More specifically, when a person enters the classroom, the system assigns an id to him/her and starts to tracking the person and providing data to the system until he/she exits the room. In the case, that users move between the corners of the classroom the service informs for the updated users' position only if a movement range is satisfied, so as to recover from the 'gittering' problem.

Gestures recognition Bridge

This service recognizes a number of predefined gestures, some of them are swipe hand left, right, up or down, clap hands, wave hand, raise the hand (the left, the right or both of them), shaking the head positive or negative, etc.

Voice recognition Bridge

This service consumes any recognized words and phrases from the ambient environment. Furthermore, each service provides a list of words or phrases that needs to be recognize (e.g. open lights, go to next/ previous slide, create a new board, etc.). When there is a match of an identified word or phrase, then the appropriate service is informed.

6.4.3 User Profile

This category consists of services which are related to the recognition and identification of the users' in the Classroom's environment. Such services interact transparently with the users, collect data from the environment and in the end identify each user. More specifically, the following services were used (i) users' profiles, (ii) users' detection, (iii) users' recognition.

User profiles service

This service is responsible to store information about users in a central location, thus enabling the creation and distribution of users' profiles across multiple services. Furthermore, it handles the personal data (e.g. users' characteristics, users' preferences, etc.) associated with the users. The service provides functionality to create, retrieve, update, and delete users' profiles. In more details other services can use this functionality (only if they have the permission) to:

a) Create a new a user profile that does not exist in the database.
b) Retrieve information regarding a specific user.
c) Update the profile of a user, if the stored data has changed.
d) Delete a profile, if that information is not needed anymore.

In order to make use of this functionality, as mentioned above the service must have the permission. The Classroom's central controller service is responsive to handling the incoming requests and give access or not. More specifically, a service can have admin access (e.g. use the entire functionality) or limited access, for example it can only be allowed to create a profile, create a profile and retrieve information, and other possible combinations.

User Detection Service

The ambient environment of the classroom is enhanced with a number of smart cameras and other sensors which are used to monitor the users. When a user enters in the room, the system keep track of him/her and starts to extract useful data (e.g. height). The service communicates with the existing

ambient facilities so as to collect these data and exploits them to adjust interaction accordingly (e.g. reachability, follow-me). Finally, the physical characteristics profile is stored in "user's profile" service and remains available to be used by other services.

User recognition service

Regarding user recognition, when a user (e.g. a teacher or a student) enters the classroom, the environment starts to track him/her and the generated data are used in order to provide contextual awareness. The service focuses on the moving patters of the users in the environment of the Classroom. It uses the data produced by the general user tracking service, processes them and extract useful features (e.g. the positions in the room where the user moved). During this phase a pattern of how the user moves in the room is exported and provided to the CognitOS Classboard.

6.3.4 Datastore service

This service offers a universal mechanism to store both document-based and binary data (e.g. documents, images, videos). Using this service, the system can easily store any generated data (i.e. documents, binary content). The Datastore is built on top of MongoDB and Minio.

- **MongoDB**: The MongoDB database is used to handle the document-based data. MongoDB is a cross-platform document-oriented database; it is a NoSQL database, which stores data in flexible JSON-like documents. It has been chosen due to the higher performance, availability and scalability which it provides compared with other databases.
- **Minio**: Minio is an object storage server. It is used in order to store data such as photos, videos, log files, backups and container images.

As already described, the framework's services use the Datastore service to store their state, as well as the data generated by the applications and the

system. For example, the state manager stores data related to the state of board (e.g. which is the active board, the order of the boards, etc.), the boards (e.g. the list of the windows and the workspaces that are existing in each board), workspaces (e.g. info about the alternative views, which windows are hosted in each view) and windows (e.g. information about the dimensions of the window, the application that is hosted in that window, etc.). The applications suite stores information related to the installed applications (e.g. configuration information, communication information, etc.). Additionally, the Datastore keeps information about the state of the various applications information and user-generated data (e.g. captured images, annotations, etc.).

6.3.4 Event federator

The Event federator enables asynchronous communication amongst the various services of the environment, giving to the services of the CognitOS Classboard the possibility of exchanging messages asynchronously. The service is built on top of the Redis Pub/Sub mechanism [120]. In particular, a service could be any or both of publisher and subscriber. Publishers post events to named channels and subscribers register their interest to those channels to receive any relevant events. The federator performs the necessary events filtering and routing from the publishers to subscribers, while it has the ability to prioritize messages in a queue before routing.

The event federator service is exploited by the services of the CognitOS Classboard and facilitates them in the communication process. The services use different channels to publish useful information (e.g. a swipe up gesture is recognized, a new application is opened, load the next board, etc.). A bunch of other services are subscribers in those channels (e.g. shortcut managers handles interaction published events, logical board manager handles events related with the board, etc.) and are informed when a new event is published in order to perform the appropriate actions.

6.4 The Applications Suite Mechanics

The CognitOS Classboard in order to support educators during a lecture and enhance the learning process, provides an application library, which consists of a number of educational applications and other useful tools. More specifically, the applications suite is divided into two categories, (i) built in applications and (ii) external applications. In order, to handle both categories an applications suite manager is developed, which stores the configuration information related to the installed applications and provides a user-interface to install new applications. It is worth noting that the application suite is one of the most important components of the system, as the Classboard would be useless without helpful and interesting content.

6.4.1 Incorporating a new educational application

As described above, the system provides an applications' suite, which consists of a number of educational applications and useful tools. In order to provide an easy way to create new content for the system (i.e. applications, services, tools) a number of specifications are defined. Following those specifications, programmers can create new applications and "install" them in the system. Simultaneously, the same policy is followed for the embedded applications; all the embedded applications of the system need to satisfy those specifications.
The specifications' list is categorized in primary and secondary specifications. The P**rimary specification** is compulsory in order to create a new application and "install" it to the system. S**econdary specifications** are optional to be implemented.
In more details, the **primary specification** that must be satisfied is that all applications should be implemented as **web-based applications**; this means that they are accessed over the network using HTTP (Hyper Text Transfer Protocol). A Web based application often runs inside a web browser and is based on the client-server architecture.

The **secondary specifications** that should be implemented are:

a. **Instance ID**: Each application should be able to launch several times in different browsers; all the created instances of the application must run independently. In order to distinguish instances of the same application (e.g. differentiate the state of each application), an instance ID should be used. The instance ID is a unique number that each web application's server assigns a specific instance of a launched application.

b. **Application's State**: As described above, it is very important for the system to have access to the state of the launched applications (directly or indirectly). This gives the capability to build a stateful system. Two different approaches are provided: (i) the application keeps the state of each client and the Classboard has no direct access to the application's state, or (ii) the application pushes the state of its active instances to the Classboard system, who is responsible to store those state objects locally. It is up to the programmer of the application to decide what policy will be followed.

Case 1: State handled by the application

In this case, the system does not have access to the state of the application, the application is responsible to handle (e.g. store, load) the state of each different instance of the launched applications. During the launching phase, the application creates an "instance id", which the board can use so as to forward any requests to that particular instance (e.g. "save state", "restore a specific state", "terminate"). With respect to state restoration/relaunch, the system passes the "instance id" as a parameter to the application's URL for the application to find the stored state and load it.

The process consists of the following steps:

Step 1: During the closing phase of an application, the system requests the "instance id" of the launched application.

Step 2: The system forwards that id to the corresponding state service to store that information locally.

Step 3: To relaunch the application with the specific state, the system passes to the application's URL (as parameter) the "instance id".

Step 4: The application is launched with the specific state.

Case 2: State handled by the system

In this case the system is responsible to keep the state of the application. During the installation phase, the programmer provides a URL (e.g. x.x.x.x/state), at run time the system by making requests to this URL is informed about the state of the application. In order to relaunch an application with a specific state, the system passes the state of the application as a parameter to the URL during the launching phase.

The process consists of the following steps:

Step 1: During the closing phase of an application, the system requests the current state of the application.

Step 2: The system forwards the state of the application to the corresponding state service in order to handle it and store it locally in the Datastore (described in section 6.3.4).

Step 3: To relaunch a specific application, the system passes to application's URL (as a parameter) the state of the application.

Step 4: The application is launched with the specific state.

c. **Forward Share – content**: In order to share the content of an application, the applications provides a corresponding URL (e.g. x.x.x.x/forward). The system by making a request to the provided location, receives a URL where the application has uploaded the shared content. The shared content can be a video file, a document, an image. Then, the Classboard is responsible to forward this URL (e.g. to the students).

d. **Provide extra functionality:** The system provides the possibility to host extra actions in the system's menu as described in section "CognitOS Classboard Menu". Those actions are accessible and can be triggered via the tools' menu by the users. Each application can provide a number of actions which extend its functionality (e.g. annotator provides undo, redo actions). During the "installation" phase, the programmer informs the system for that extra functionality by providing a URL (e.g. x.x.x.x/actions), where the system can find information for those actions, i.e. (i) the name of the action, (ii) a url to trigger this action, (iii) a URL to an icon related to this action.

e. **Testing mechanism**: Each service can implement its own test checking mechanism. This mechanism checks if all of the provided functionality is working correctly based on the specifications of the service. If the service implements a testing mechanism, it should provide a URL (e.g. x.x.x.x/test) during the installation. The system by making a request to the provided URL triggers the test checking mechanism and receives back information about the service state (e.g. "on-line" if the service running as it is expected, "off-line" if the service could not be reached, or "pay-attention" if something during the testing goes wrong).

6.4.2 Applications Suite Manager

As it is implied by its name, it manages the applications' suite. More specifically, the applications manager is split into two parts, namely backend and frontend. The backend handles all the data of the installed applications and is responsible for providing the corresponding content to the system. On the other hand, the frontend provides a user interface to install new applications in the system and to update the information of the existing applications.

Applications Suite Manager Backend

The backend service of the applications' manager is responsible for handling and storing all the configuration data related to the installed applications. During the "installation phase" the programmer provides all the needed configuration information for an application; in the next step, the backend gets that info and stores it in the system's data store. This list of the "installed" applications and the configuration information of each application are always available and accessible via the backend; such data can be used (e.g. displayed) by the frontend service of the applications suite manager or from the Classboard. Additionally, the CognitOS Classboard makes use of those applications and provides them to the educator and the students.

Applications Suite Manager Frontend

The frontend part of the applications' manager provides (i) an overview of all the "installed" applications (e.g. configuration information) and (ii) a user interface to embed new applications in the system. It also allows the programmer to update the information of the installed applications. During the "installation" phase, the programmer provides a number of info such as:

a. **Name:** The name of the application.
b. **URL**: The location where the application is served.
c. **Width/ Height**: Provide information about the size limitations of the application (e.g. minimum width/height, maximum width/height).
d. **Description**: A short description of the application.
e. **State URL**: The endpoint from where the CognitOS Classboard can get information about the current state of the launched application.
f. **Forward Share URL**: Get the content to be forward.
g. **Actions URL**: Get the information about the extra actions that served via tools menu.
h. **Testing URL**: Get information about the state of the service.

In the list above, only the **URL** where the application is served is obligatory; the remaining information is optional. It can be used to extend the

functionality of the system but an application could be launched without the provision of those info.

Chapter 7

Evaluation

This chapter describes the evaluation process of the CognitOS Classboard system; two different approaches were used, (i) cognitive walkthrough on the designed prototypes of the user interface and (ii) user-based evaluation of the implemented system. These two processes were conducted at the Human-Computer Interaction Laboratory (HCI) of the Institute of Computer Science of the Foundation for Research and Technology - Hellas (ICS-FORTH).

7.1 Cognitive Walkthrough

A cognitive walkthrough evaluation experiment of the CognitOS Classboard was conducted with the participation of five (5) User Experience (UX) experts. The goal was to assess the overall concept and its potential, identify any unsupported features and uncover potential usability errors by noting the comments and general opinion of experts before planning a large-scale user-based experiment. The cognitive walkthrough is an inspection method for evaluating the design of a user interface, with special attention to how well the interface supports exploratory learning, i.e., first-time use without formal

training [121]. Its purpose is to identify whether or not a user can easily carry out specific tasks within a given system. The nature of the cognitive walkthrough defines it as one of the fastest forms of usability testing, since the user is required to only carry out small tasks. The method can be used prior to development and during the design phase of a system.

7.1.1 Process

During the experiment, a conductor helped the users (UX Experts) to work through a series of tasks and asked them a set of questions to understand how simple is to interact with the CognitOS Classboard workspaces. Using a custom observation grid the conductor was keeping notes regarding whether they tried and achieved the desirable outcome, whether they noticed that the correct action is available to them, and if the action is perceived to be associated with the expected outcome. Finally, when the experiment was concluded, the conductor recorded the steps that seemed to trouble the users, as well as the issues that were identified.

7.1.2 Results

The most notable issues that were identified from the evaluation process, along with the proposed solutions, are listed below:

Issue 1: Given that a user can launch multiple applications on demand in the same workspace, there is a possibility that the available space will not suffice for all of them to be simultaneously visible.

Solution: The CognitOS Classboard will: (a) support both automatic (i.e. following a Least-Recently used policy) and on-demand applications' minimization to save some screen real-estate and (b) recommend to the user to launch the new application either in an adjacent smaller workspace (with enough space) or create an entirely new one to place it.

Issue 2: The CognitOS Classboard workspaces can potentially overwhelm educators in creating them (e.g. some might over-use tablet and desktop configurators and create a new workspace for every new application).

Solution: The accompanying tablet and desktop configurator will feature a creation wizard to guide the educators during the initialization of their first few workspaces, while it will also assess their actual requirements by monitoring the board's real-time use (i.e. how many times did the educator changed a workspace, which are the most frequently used workspace and applications) to personalize the relevant recommendation algorithms.

Issue 3: The educators very often leave their workspace and move around the podium or classroom to emphasize their points and communicate more effectively with their audience. In that case, they should be able to control the system remotely.

Solution: The CognitOS Classboard will integrate various interaction modalities (e.g. gestures, voice commands, remote mouse) through which educators will be able manipulate it (e.g. change workspace, launch a new or close a running application) from a distance.

Issue 4: The educators more than often need to update the way they teach to accommodate a different pedagogy or react on unforeseen events (e.g. an interesting inquiry of a student). Therefore, they should be able to freely move the applications around the board, replace parts of the board or even mark some of them as unmovable (i.e. pinned).

Solution: The workspaces feature of the CognitOS Classboard will be further enhanced with advanced functionality such as a "workspace-in-a-workspace" special type of window, the ability to manipulate workspaces as if they were simple windows (e.g. pin, move, close, migrate), etc.

7.2 User Based Evaluation

After the completion of the cognitive walkthrough process, the system was redesigned and prepared for a user-based evaluation. User-based evaluation is the most fundamental usability method and is irreplaceable. This method helps to understand what works and what does not e.g. in an interface, in a system, in an application, etc.; by observing people using it. When the right participants attempt realistic activities, the researchers gain qualitative

insights into what is causing trouble to the users and helps to determine how to improve the design. Additionally, user-based evaluation provides ways to measure the percentage of tasks that users complete correctly as a way to communicate e.g. an interface's, a system's, an application's overall usability [122].

Compliance with ethical standards: During the evaluation period, only the absolutely necessary data was collected and processed through a "pseudonymization" process, while participants' identities will remain confidential and not be revealed. More specifically, the following data was recorded:

- Personal data (i.e. name, age, gender, and contact details)
- For each task, info about success or failure completion, how long it took to complete it, the number of errors and a short description of each error, etc.
- Comments, reactions, and user responses to questions and questionnaires.

In the context of the evaluation process, both the European Union (EU) regulation on General Data Protection (GDPR; 679/2016) and the Greek Applicable Law 4624/2019 have been properly taken into account and approval has been granted by the FORTH Ethics Committee. In addition, participants were given Information about the nature of the evaluation and all aspects of participation, and a consent form was signed, which has been prepared in collaboration with the Data Protection Offer (DPO) of FORTH.

7.2.1 Research Questions

The user-based evaluation experiment was conducted inside the simulation space of the Intelligent Classroom, with the participation of 10 users which were asked to play the role of a University professor. The experiment did not entail the participation of students, however the users were asked to complete a series of tasks, as if there were students attending their class. The goal of the study was to identify any unsupported features, uncover potential usability

issues and assess the users' overall experience before proceeding with a study including the active participation of students. In particular, the following research questions were investigated:

Research Question 1: Is the CognitOS Classboard usable?

Research Question 2: Is the integrated functionality useful?

Research Question 3: Do users exhibit interaction related issues?

7.2.2 Participants

The intended users of the system are both adults (educators/presenters) and children (students), however this study focuses on getting insights by observing educators/presenters interacting with the system. To this end, a total of ten (10) adults participated to the experiment; all of them had experience either in making presentations or teaching students. This number of users is appropriate for preliminary evaluations, as it can identify important usability problems before proceeding to large scale experiments [122].

The selected users meet different characteristics (e.g. age, gender, teaching/presentation experience). In particular, from the total of 10 users, the 60% were females, and the 40% were males. Moreover, users were selected in order to include an as wide as possible range of ages; three of the users were in their twenties, four users were between 30 and 40 years old, and three were older than 40 years old. Additionally, their presentation or teaching experience varied so as to record the opinion and comments of different types of users (e.g. users having a teaching experience of more than 10 years, versus users with a teaching experience less than a year) (Figure 53).

None of the participants had experience with Interactive Whiteboards. However, the majority of them (90%) use projectors to display their presentations, while 80% of them also use the traditional Whiteboard (Figure 54) so as to write important information or make notes, etc. For future evaluation experiments, we intend to include users who have experience with commercial Whiteboards, so as to record their comments and opinion as well.

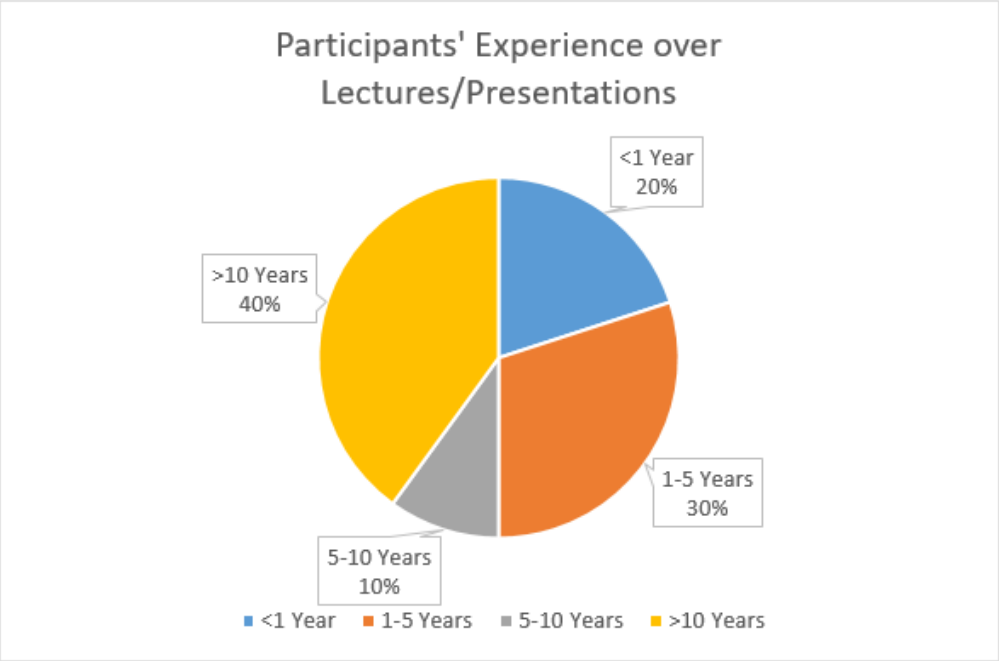


Figure 53: Participants' experience over lectures / presentations.

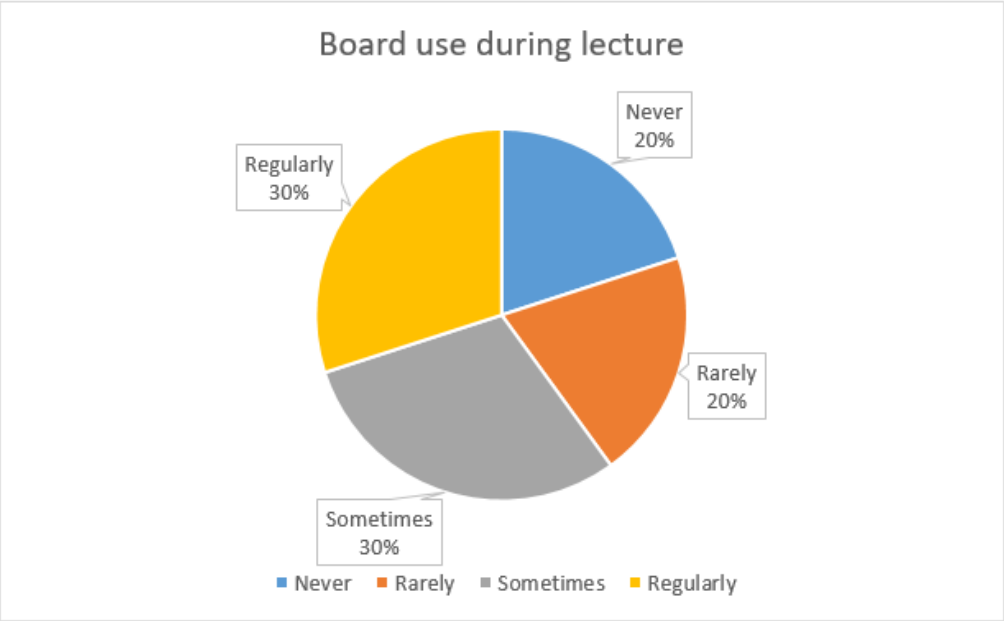


Figure 54: Use of board during lecture.

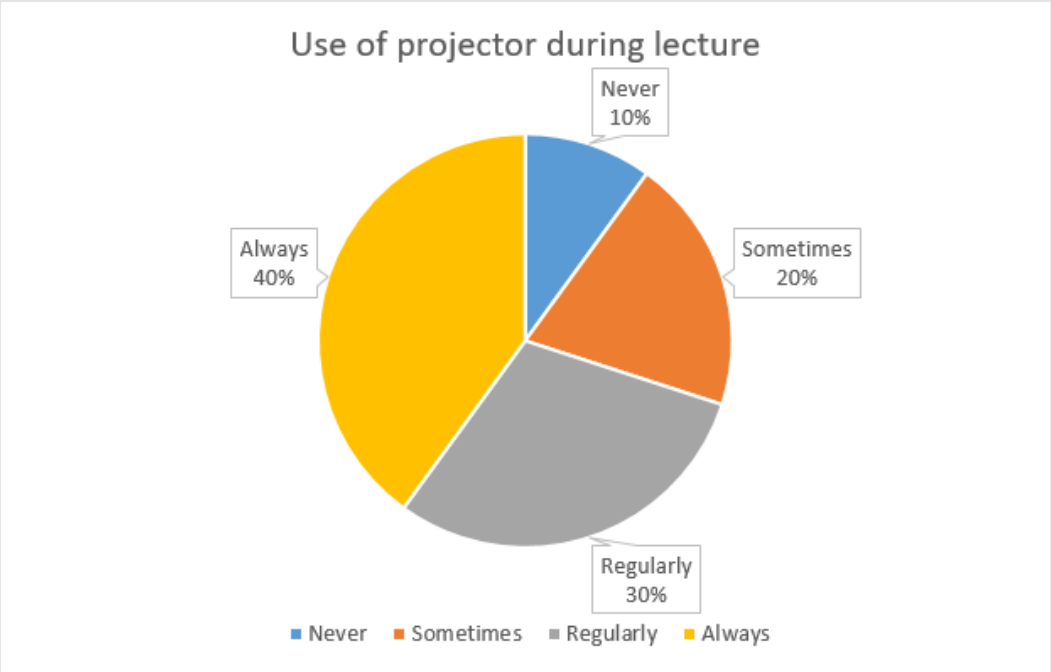


Figure 55: Use of projector during lecture.

7.2.3 Data Collection

In order to explore the aforementioned research questions, a pre-evaluation questionnaire, a custom observation grid and two post-evaluation questionnaires were used.

Pre-evaluation Questionnaire

A pre-evaluation questionnaire has been used in order to collect participants' demographics and other user-related information:

- The age and gender of the participants
- The teaching / presentation experience of the participants, both in terms of duration (i.e. less than a year, 1-5 years, 5-10 years, more than 10 years) and frequency (i.e. weekly, monthly, one over six months, once a year)
- Frequency and reason of use of the traditional classroom board
- Experience with Interactive Whiteboards
- Experience with projected presentations and experience with presentation gadgets (e.g. presenters)

Observation Grid

A custom observation grid was used to collect qualitative and quantitative information:

- The overall duration of the running test
- The comments of users' during the test
- The number and type of errors made during each task
- The number of hints/helps given during each task
- The time it took users to complete each task.

Post-evaluation Questionnaires

Two (2) questionnaires were used, aiming to reveal the usability of the system, as well as the overall user satisfaction. The first one was the System Usability Scale (SUS) [123], a subjective satisfaction questionnaire which included 10 questions, with five response options for respondents, from strongly agree to strongly disagree. The SUS questionnaire was translated in Greek, according to [124], so as to be more comprehensible by the participants. The second one, was a custom questionnaire, which included 10 questions the about users' overall impression regarding the CognitOS Classboard:

1. How did you find the idea of an Intelligent Board that aims to enhance teaching or presentations in general?
2. Do you believe that a board of such dimensions (wall-to-wall) would be useful for teaching or giving presentations?
3. Would you like a similar board for your presentations? How often would you use it?
4. How did you find the interaction with the board?
5. What do you think of the "Follow-me" functionality?
6. What do you think of the "Summon" functionality?
7. What was the most difficult thing that you had to do during the execution of the scenario?
8. What did you like the most?
9. Is there something that you did not like, and that you would prefer to change?
10. Is there any functionality that you would expect to find, but it was not integrated in the system?

7.2.4 Procedure

The study included four stages: (i) preparation, (ii) introduction, (iii) running the test and (iv) debriefing [122]. In order to make participants feel comfortable and ensure that the experiment progressed as planned, a facilitator was responsible for orchestrating the entire process and assisting the users when required, while an observer was responsible for recording information in custom observation grid. Additionally, a member from the experimental team was available, in order to manage any technical difficulties.

Preparation

In the preparation stage, all the appropriate actions were completed before the arrival of the participants, in order to make sure that the experiment would be conducted without problems. In particular, the projectors were turned-on, the system was initialized, and the room A/C was set at a comfortable temperature. Moreover, the material required for the experiment (i.e., pre- and post- evaluation questionnaires, observation grid, and scenario) was prepared and organized in a way that permits both the facilitator and observer to manipulate it easily.

Introduction

During the introduction stage, the facilitator welcomed the participants (Figure 56), gave a brief explanation of the purpose of the experiment, and highlighted the importance of their participation. Next, the concept of the Intelligent Classroom was described, followed by a short introduction of the CognitOS Classboard and its main concepts (i.e. Walls, Boards, Workspaces). Next, a demonstration of the interaction paradigm followed, including three simple tasks performed by the facilitator (i.e. launch menu, launch an application, summon application to side wall).

After that, the facilitator clarified that the users' personal data would be anonymized, as described in the Informed Consent Form, which they were asked to sign it prior to their participation in the experiment. Finally, the

facilitator told them that they could stop the experiment whenever they wanted and encouraged them to follow the Think-Aloud protocol [125]. According to this protocol, the participants were asked to verbalize their thoughts as they moved through the User Interface so as to provide valuable feedback regarding the difficulties they face during the experiment as well as their overall impression of the system.

Figure 56: Introduction stage of evaluation.

Running the test

After the introductory phase, the participants were asked to follow a scenario including ten (10) tasks, which was projected on a side wall, one task at a time. The users could either read the tasks by themselves or ask the facilitator to narrate them. During that process, the facilitator refrained from interacting with the participants and did not express any personal opinion regarding whether they were doing well or poorly. The only exception to this rule was made when a participant was having some difficulty in completing a task; at this point, the facilitator intervened to provide a helpful tip. At the same time the observer was keeping notes on the custom observation grid, while a member of the experimental team was standing by in order to manage any technical difficulties.

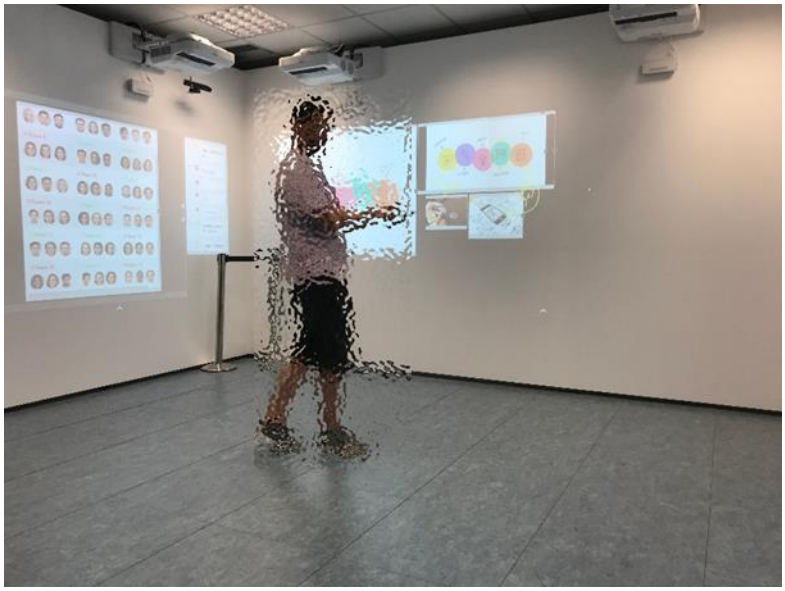


Figure 57: Running the test.

Debriefing

After the experiment was completed, the facilitator debriefed the users according to a set of 10 questions (section 7.2.3), in order to investigate their overall impression and record any thoughts or suggestions that they had to make. In the end, the users were asked to fill in the SUS questionnaire and the custom-made questionnaire by themselves. Finally, the facilitator thanked the users for their participation and offered them some treats in order to express her gratitude.

7.2.5 Results

In this section, the results of the user-based evaluation are presented in relation to the research questions that have been mentioned previously.

Research Question 1: Is the CognitOS Classboard usable?

The evaluation results indicated that in general the users did not encounter major problems and there were no failures. In more detail, the distinct errors that users made were 13 (Table 8), while the total number of erroneous interactions was 29. These errors occurred due to minor usability issues that were reveal, which can be easily eliminated for the next version of the CognitOS Classboard in order to help users avoid such missteps.

Table 8: The frequency of the errors made by the users.

Errors made by the users	**Frequency**
User did not understand how to initiate Capture functionality	5
User tried to scroll a window from its interior instead of its scrollbar	4
User tried to resize the area to be Captured	3
Navigation between Boards confused the user	3
User tried to perform a gesture on top of other windows	3
User did not understand that the captured image was successfully shared to the students	3
User accidentally erased annotations by selecting the eraser tool	2
User was confused with clear all functionality of Annotate	1
User did not understand that the active Board changed	1
User did not understand which Workspace or Board was active since the visual cues on the paging component were misleading	1
User expected to find the share functionality in the Application-specific menu, and not in the tools	1
User was unable to locate the Schedule application	1
User did not understand immediately how to pin a window	1

As depicted in the following chart Figure 58, 30% of the users made only one error, 50% made 2-4 errors, while the remaining 20% made 5-6 errors. The total number of errors per user is relatively small, considering that they had to complete quite demanding scenario consisting of 10 tasks and that it was their first time interacting with the system. The analysis of the evaluation findings revealed that none of the users repeated the same errors, which means that they became familiar with the system during their interaction with it. In fact, some users pointed out this aspect by themselves: “I did this task before, now I know how to do it! It is easy”, “Previously I was slightly confused, because it was the first time, but now I know what to do! It is clear!”. Finally, the chart Figure 58 shows that in general the users received less hints than the errors they made, while the observation notes reveal that all of them quickly

completed the required task, which indicates that the system was able to help them recover from errors easily.

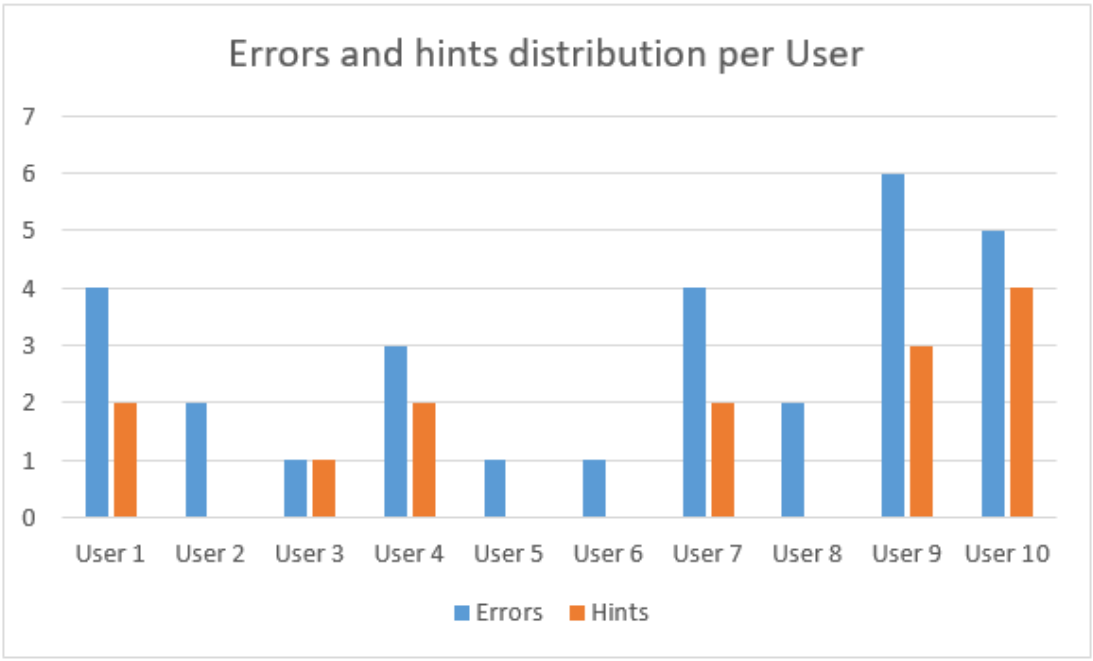


Figure 58: Errors and hints distribution per user.

Furthermore, the SUS questionnaire has been used in order to measure the perceived usability. In general, participants' responses were positive, as it is visualized in Figure 60. The overall score was 83, which is higher than the average SUS score (68). Moreover, according to the curved grading scale for SUS [126] (Figure 59), the CognitOS Classboard received overall an A score. Further investigation of the individual SUS scores reveals that 50% of the users rated it with an A+ (ranging from 85 to 100), the 30% of the users rated it with an A and A- (ranging from 80 to 82.5), while the remaining 20% rated it with a C (ranging from 65 to 70).

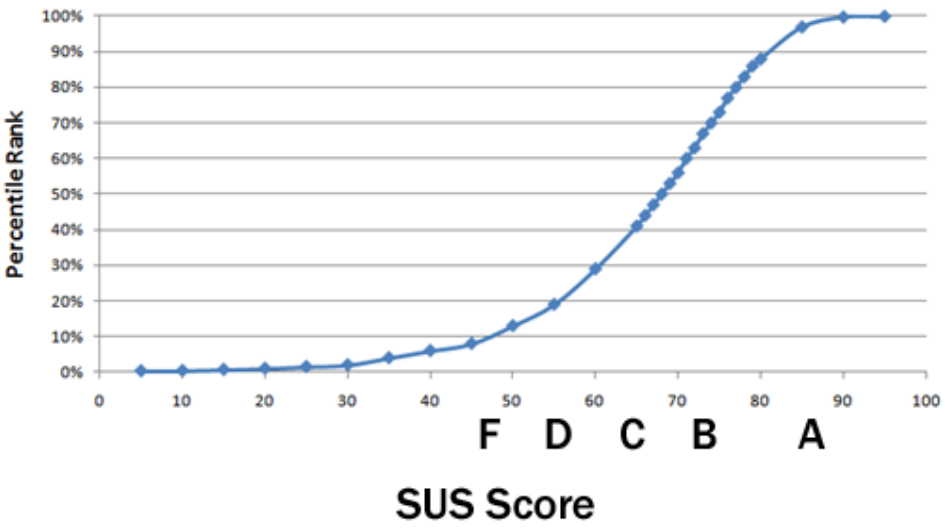


Figure 59: Curved grading scale for SUS [127].

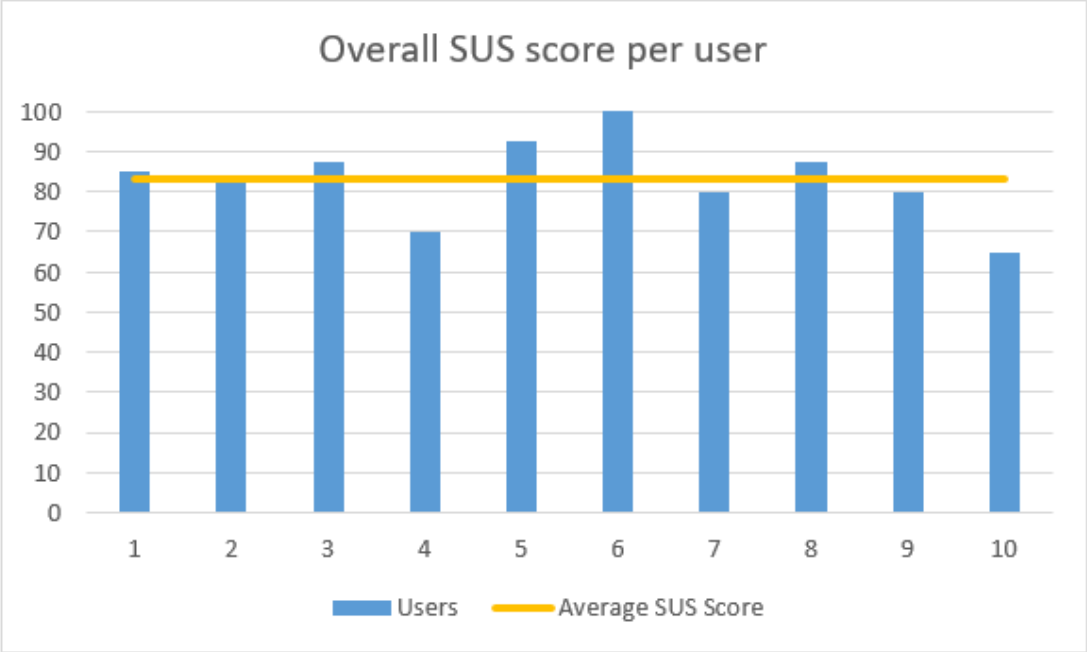


Figure 60: Overall SUS score per user.

Research Question 2: Is the integrated functionality useful?

In order to understand whether the integrated functionality is useful, we studied the comments made by the users during the evaluation as well as their responses to the custom post-evaluation questionnaire. It was revealed that all users found the system extremely useful and responded that they will would definitely use it, if it was available. For example, one participant said that "The

system is excellent, I would like to use it during my lecture", a second one stated that "The idea is brilliant, I it would be very beneficial for educational purposes", while another one expressed that "It is very useful system, it provides many facilities that can enhance presentations".

Furthermore, the user responses to the questions 5,6 and 8 of the post-evaluation questionnaire (Section 7.2.3) revealed useful insights regarding the Follow-me, Summon, Capture and Annotate functionalities. In more detail, all users were particularly fond of the Summon functionality; all of them recognized that the ability to summon an application located in a far position is of outmost importance. Indicatively, some of the users' comments include: "It is the best and most useful feature of the system", "Ohhh! That functionality is very good", "It requires less effort than to drag windows around the board". The Follow-me functionality impressed fewer participants. In more detail, 60% of them found this feature necessary and useful, 20% thought that it would be more essential for larger boards, while the remaining 20% believed that it is fun but they questioned its usefulness. The participants that were able to visualize the potential of the Follow-me functionality, made some very positive comments, such as: "It is very useful, I like that the window is moving based on my location, so I could interact with the window faster than in usual", "It is great, very useful, you could make the window you want to emphasize to follow you". However, based on these findings, we believe that in a large-scale experiment that will take place in a room with a larger board, the full potential of this functionality will be revealed.

Finally, the integration of the Annotate and Capture functionalities received positive feedback, such as "I enjoy the annotation functionality, I could use it all the time to emphasize or highlight content", "The combination of capture and annotation functionalities is extremely useful, one can easily combine digital content with handwritten annotation and share to the audience".

Research Question 3: Do users exhibit interaction related issues?

During the evaluation it was observed that the users were confused regarding when to press the pen button while interacting with the system. In general, the pen acts as a mouse cursor, while pressing button activates the right click functionality. The latter is required only in two cases: (i) launching the menu, and (ii) performing a gesture.

So, despite the fact that during the introduction made by the facilitator, the interaction paradigm was explained to the users, two types of errors were revealed: (i) the users did not press the pen button when required, (ii) the users pressed the button every time. However, it was observed that the more they interacted with the system, the more accustomed they became with the interaction paradigm.

Another issue that was revealed is the fact that the users were not sure where to perform a gesture. In particular, 4 users tried to make the gesture over an application, instead over an empty space of the board. However, this is something that can be learnt by providing some support to novice users (tutorial, on-the-fly tips), as Window users have learned that right clicking on an empty desktop area reveals a different menu than right clicking on an application icon.

Another interaction related issue that this study aimed to investigate, was whether the user would understand how to navigate amongst the multiple boards. Interestingly, 40% of the users at their first attempt used the swipe up area on the bottom of the board to reveal the next one, 20% used the menu and the remaining 40% used the pagination component located on the top of the board. The tallest users immediately used the pagination component, while the ones that could not reach it sought for an alternative. In any case, the navigation amongst the available boards were possible and easily performed by all users.

Finally, another important aspect that we wanted to examine was how the users would try to move applications amongst boards. The findings reveal that, 70% of the users instinctively used the summon functionality (that was already

demonstrated by the facilitator during the introductory phase), 20% of the users tried by dragging and dropping the window on the side wall, while one user used the board manager. Additionally, one user suggested that one way to accomplish this task would be with the Follow-me functionality. It was quite satisfactory to observe that the system supports all of the approaches that the users selected to follow.

Chapter 8

Conclusion and Future Work

This work equips the **Intelligent Classroom Board** of ICS-FORTH with a sophisticated framework, named **CognitOS Classboard** (Figure 61), which takes advantage of (i) the intelligent facilities offered by the environment (e.g. user localization service) and (ii) the amenities offered by the wall-to-wall board in order to enhance educational activities and provide educator- and student- oriented experiences.

The educator is equipped with a unified working environment for the large interactive board that offers a natural and effortless interaction paradigm, allowing the effective orchestration of teaching assets (e.g. use different workspaces to place different types of applications) and the presentation of rich digital material to the students. Additionally, course preparation before class is supported via a desktop application that allows educators to initialize the contents of the board with the appropriate material in advance.

Aiming to be as less invasive as possible in the classroom environment and in tone in with the already standardized processes of lecturing and teaching, CognitOS Classboard supports multimodal interaction and follows well-

established interaction metaphors (e.g. writing on a digital board should be similar to writing on a plain whiteboard).

Figure 61: A snapshot of CognitOS Classboard.

From the students' perspective, aiming to create highly engaging and fascinating learning experiences, the CognitOS Classboard, apart from offering access to useful educational applications provides mechanisms capable of transforming the classroom into an immersive environment on demand.

As indicated by the evaluation study in general the users did not encounter major problems and there were no failures. In particular, the system proved to be very usable and useful, while according to the curved grading scale for SUS, the CognitOS Classboard received overall an A score. The minor issues that were revealed during the evaluation, are going to be resolved for the next version of the system, while we are going to take into consideration several suggestions made by the users so as to improve the overall user experience.

CognitOS Classboard has been initially designed to extend the modern version of a blackboard (i.e. interactive smart board), but after evaluating the system with end-users, we identified that the system has also the potential to evolve into a general presentation tool that could be used anywhere (e.g. meeting rooms, auditoriums, conference venues). Subsequently, potential future improvements, apart from the evaluation findings, include: (i) the extensive

interoperation with the various services of the Intelligent Classroom environment so as to provide a better user experience (both to educators and students), (ii) the improvement of the items placement algorithm to further assist educators in better organizing their teaching assets, (iii) the exploitation of recurring user interaction patterns (e.g. how the user usually set ups the board, organization of the available workspaces) so as to provide tips and suggestions (e.g. suggest appropriate layouts to set up the board, place launched applications at specific positions on the board, etc.), (iv) the enhancement of supported gestures and shortcuts, (v) the development of a monitoring mechanism to guarantee a proper Quality of Service (QoS) from the components that comprise the system and the external applications, and finally (vi), the introduction of more educational applications and tools to support lecturing/presentation.

Bibliography

[1] V. L. Tinio, *ICT in Education*. e-ASEAN Task Force, 2003.

[2] W. I. O'Byrne, K. Houser, R. Stone, and M. White, "' Digital storytelling in early childhood: Student illustrations shaping social interactions': Corrigendum.," 2019.

[3] E. Papadimitriou, A. Kapaniaris, D. Zisiadis, and E. Kalogirou, "Digital storytelling in Kindergarten: An Alternative tool in children's way of expression.," *Mediterr. J. Soc. Sci.*, vol. 4, no. 11, p. 389, 2013.

[4] R. C. Schank, "Active learning through multimedia," *IEEE Multimed.*, vol. 1, no. 1, pp. 69–78, 1994.

[5] B. Robin, "The power of digital storytelling to support teaching and learning," *Digit. Educ. Rev.*, no. 30, pp. 17–29, 2016.

[6] H. Kanematsu and D. M. Barry, *STEM and ICT education in intelligent environments*. Springer, 2016.

[7] E. Milkova, S. Pekarkova, and A.-B. M. Salem, "Information and communication technology in education-current trends," in *MATEC Web of Conferences*, 2016, vol. 76, p. 04022.

[8] J. King and J. South, "Reimagining the role of technology in higher education: A supplement to the national education technology plan," *US Dep. Educ. Off. Educ. Technol.*, 2017.

[9] "The dawn of artificial intelligent schooling." https://inews.co.uk/news/education/dawn-ai-school-future-classroom-327753 (accessed Apr. 14, 2020).

[10] "Smart Classroom Tools For Smart Teachers | Education Technology," *Cybernetyx*, Mar. 22, 2019. https://en.cybernetyx.com/smart-tools-for-education/ (accessed Apr. 14, 2020).

[11] “Classroom-Technology | Solutions | AVer Global.” https://www.aver.com/solution/classroom-technology?_ga=2.127310861.1818085406.1580216138-1902838730.1580216138 (accessed Apr. 14, 2020).

[12] “The classroom of the future: back to school 2026,” *TheSchoolRun*. https://www.theschoolrun.com/classroom-of-the-future (accessed Apr. 14, 2020).

[13] M. Saville, K. Beswick, and R. Callingham, “The Use of Interactive Whiteboards in Education: Opportunities and Challenges,” in *The Future of Educational Research*, Brill Sense, 2014, pp. 203–216.

[14] R. Parmet, “Jumpstart Classroom Engagement with Interactive Whiteboard Technology,” *Parmetechinc*, Mar. 18, 2019. https://www.viewsonic.com/library/education/jumpstart-engagement-interactive-whiteboard (accessed Apr. 14, 2020).

[15] S. Bennett and L. Lockyer, “A study of teachers’ integration of interactive whiteboards into four Australian primary school classrooms,” *Learn. Media Technol.*, vol. 33, no. 4, pp. 289–300, 2008.

[16] J. Aguilar, P. Valdiviezo, J. Cordero, and M. Sánchez, “Conceptual design of a smart classroom based on multiagent systems,” in *Proceedings on the International Conference on Artificial Intelligence (ICAI)*, 2015, p. 471.

[17] K. Alshahrani and M. Ally, “Smart classrooms in the context of technology-enhanced learning (TEL) environments: a holistic approach SALAH AL-SHARHAN,” in *Transforming Education in the Gulf Region*, Routledge, 2016, pp. 216–242.

[18] “Interactive Whiteboard Market by Screen Size, Technology, End User, and Geography | COVID-19 Impact Analysis | MarketsandMarkets™.” https://www.marketsandmarkets.com/Market-Reports/interactive-whiteboard-market-51706837.html?gclid=Cj0KCQiAkePyBRCEARIsAMy5ScszsazPnmoqc6Wka

stJ9ZcvWnkEpm9n6NzCdm6Qxz_hqzt8UBIr6bkaAm8UEALw_wcB (accessed Apr. 14, 2020).

[19] J. Sarsa and R. Soler, "Special features of Interactive Whiteboard software for motivating students," *Int. J. Inf. Educ. Technol.*, vol. 1, no. 3, pp. 235–240, 2011.

[20] S. Hennessy, "The role of digital artefacts on the interactive whiteboard in supporting classroom dialogue," *J. Comput. Assist. Learn.*, vol. 27, no. 6, pp. 463–489, 2011.

[21] E. Manny-Ikan, O. Dagan, T. Tikochinski, and R. Zorman, "[Chais] Using the Interactive White Board in Teaching and Learning–An Evaluation of the SMART CLASSROOM Pilot Project," *Interdiscip. J. E-Learn. Learn. Objects*, vol. 7, no. 1, pp. 249–273, 2011.

[22] D. J. Cook, J. C. Augusto, and V. R. Jakkula, "Ambient intelligence: Technologies, applications, and opportunities," *Pervasive Mob. Comput.*, vol. 5, no. 4, pp. 277–298, 2009.

[23] E. Aarts and R. Wichert, "Ambient intelligence," in *Technology guide*, Springer, 2009, pp. 244–249.

[24] A. Aztiria, A. Izaguirre, and J. C. Augusto, "Learning patterns in ambient intelligence environments: a survey," *Artif. Intell. Rev.*, vol. 34, no. 1, pp. 35–51, 2010.

[25] J. C. Augusto, "Ambient intelligence: the confluence of ubiquitous/pervasive computing and artificial intelligence," in *Intelligent Computing Everywhere*, Springer, 2007, pp. 213–234.

[26] L. R. Winer and J. Cooperstock, "The 'intelligent classroom': Changing teaching and learning with an evolving technological environment," *Comput. Educ.*, vol. 38, no. 1–3, pp. 253–266, 2002.

[27] M. Montebello, "Technological Aspect," in *The Ambient Intelligent Classroom*, Springer, 2019, pp. 49–70.

[28] M. Korozi, A. Leonidis, M. Antona, and C. Stephanidis, "LECTOR: Towards Reengaging Students in the Educational Process Inside Smart

Classrooms," in *International Conference on Intelligent Human Computer Interaction*, 2017, pp. 137–149.

[29] R. A. Ramadan, H. Hagras, M. Nawito, A. El Faham, and B. Eldesouky, "The intelligent classroom: Towards an educational ambient intelligence testbed," in *2010 Sixth International Conference on Intelligent Environments*, 2010, pp. 344–349.

[30] R. A. Ramadan, "Leveraging RFID technology for intelligent classroom," in *2009 4th International Design and Test Workshop (IDT)*, 2009, pp. 1–6.

[31] K. Alshahrani and L. Cairns, "Managing the change during e-learning integration in higher education: a case study from Saudi Arabia," in *Transforming Education in the Gulf Region*, Routledge, 2016, pp. 195–205.

[32] M. Montebello, "The AmI Classroom from a Technological Perspective."

[33] M. Korozi *et al.*, "Shaping the Intelligent Classroom of the Future," in *International Conference on Human-Computer Interaction*, 2019, pp. 200–212.

[34] C. Savvaki, A. Leonidis, G. Paparoulis, M. Antona, and C. Stephanidis, "Designing a technology–augmented school desk for the future classroom," in *International Conference on Human-Computer Interaction*, 2013, pp. 681–685.

[35] E. Stefanidi, D. Arampatzis, A. Leonidis, and G. Papagiannakis, "BricklAyeR: A Platform for Building Rules for AmI Environments in AR," in *Computer Graphics International Conference*, 2019, pp. 417–423.

[36] C. Ardito, R. Lanzilotti, M. F. Costabile, and G. Desolda, "Integrating traditional learning and games on large displays: an experimental study," *J. Educ. Technol. Soc.*, vol. 16, no. 1, pp. 44–56, 2013.

[37] J. Gage, *How to use an interactive whiteboard really effectively in your secondary classroom*. David Fulton Publishers, 2013.

[38] "Epson education solutions." https://www.epson.eu/verticals/business-solutions-for-education (accessed Mar. 19, 2020).

[39] “ViewSonic Education Solutions – Digital Whiteboards in the Classroom.” https://www.viewsonic.com/education/ (accessed Mar. 19, 2020).

[40] “EZWrite 4 Series | BenQ Display Solutions.” https://business-display.benq.com/en/findproduct/ifp/software/e-z-write.html (accessed Mar. 19, 2020).

[41] “MagicIWB | Software Solutions,” *Samsung Display Solutions*. https://displaysolutions.samsung.com/solutions/signage-solution/magiciwb (accessed Mar. 19, 2020).

[42] “Oktopus - Value-added Software for IFP | AVer Global.” https://presentation.aver.com/line/interactive-flat-panel/Oktopus (accessed Mar. 19, 2020).

[43] “AVer Interactive Flat Panel & Control Box | AVer Global.” https://presentation.aver.com/lines/interactive-flat-panel (accessed Mar. 23, 2020).

[44] M. Korozi, S. Ntoa, M. Antona, and C. Stephanidis, “Intelligent working environments for the ambient classroom,” in *International Conference on Universal Access in Human-Computer Interaction*, 2011, pp. 381–390.

[45] B. Cao, M. Esponda, and R. Rojas, “The Use of a Multi-Display System in University Classroom Lectures,” in *Proceedings of the 2016 ACM on Interactive Surfaces and Spaces - ISS ’16*, Niagara Falls, Ontario, Canada, 2016, pp. 427–432, doi: 10.1145/2992154.2996793.

[46] B. Cao, M. Esponda-Argüero, and R. Rojas, “DEVELOPMENT AND EVALUATION OF A CLASSROOM INTERACTION SYSTEM,” p. 8, 2016.

[47] “SMART Learning Suite Software - Media Technology - DEKOM.” https://www.dekom.com/en/media-technology/product/ (accessed Apr. 14, 2020).

[48] “Formative Assessments, Learning Games & More // SMART Learning Suite - SMART Technologies.” https://www.smarttech.com/en/Products/Education-Software/SMART-Learning-Suite (accessed Mar. 19, 2020).

[49] “ActivInspire Lesson Delivery Software | Promethean,” *Promethean World*. https://www.prometheanworld.com/products/lesson-delivery-software/activinspire/ (accessed Apr. 14, 2020).

[50] “MimioStudio Classroom Software For Interactive Whiteboard Lessons,” *Boxlight*. https://mimio.boxlight.com/mimiostudio-classroom-software/ (accessed Mar. 19, 2020).

[51] “MultiTaction Software - The Ultimate Interactive Experience,” *MultiTaction*. https://www.multitaction.com/multitaction-software/ (accessed Mar. 19, 2020).

[52] “Lü - Interactive Playground,” *Lü - Interactive Playground*. https://www.play-lu.com/ (accessed Mar. 19, 2020).

[53] “Enhancing classroom lectures with digital sliding blackboards | ACM SIGCSE Bulletin.” https://dl.acm.org/doi/10.1145/1026487.1008054 (accessed Mar. 20, 2020).

[54] “Interactive whiteboard software - OneScreen Annotate.” https://www.onescreensolutions.com/en/product/annotate (accessed Mar. 19, 2020).

[55] “Annotate | Mobile Interactive Whiteboard, Student Response System,” *Annotate*. https://annotate.net/ (accessed Mar. 19, 2020).

[56] “ActivInspire Lesson Delivery Software | Promethean,” *Promethean World*. https://www.prometheanworld.com/products/lesson-delivery-software/activinspire/ (accessed Mar. 19, 2020).

[57] H. Plattner, C. Meinel, and U. Weinberg, *Design-thinking*. Springer, 2009.

[58] R. F. Dam and Y. S. Teo, “5 Stages in the Design Thinking Process,” *The Interaction Design Foundation*. https://www.interaction-design.org/literature/article/5-stages-in-the-design-thinking-process (accessed May 03, 2020).

[59] J. M. Carroll, “Five reasons for scenario-based design,” *Interact. Comput.*, vol. 13, no. 1, pp. 43–60, 2000.

[60] A. Leonidis, D. Arampatzis, N. Louloudakis, and C. Stephanidis, “The AmI-Solertis System: Creating User Experiences in Smart Environments,” presented at the The 13th IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, 2017.

[61] A. Ntagianta, M. Korozi, A. Leonidis, M. Antona, and C. Stephanidis, “CognitOS: A Student-Centric Working Environment for an Attention-Aware Intelligent Classroom,” in *Proceedings of the 20th International Conference on Human-Computer Interaction*, 2018, p. (submitted).

[62] Z. Sheng, S. Yang, Y. Yu, A. Vasilakos, J. Mccann, and K. Leung, “A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities,” *IEEE Wirel. Commun.*, vol. 20, no. 6, pp. 91–98, 2013.

[63] D. Bandyopadhyay and J. Sen, “Internet of things: Applications and challenges in technology and standardization,” *Wirel. Pers. Commun.*, vol. 58, no. 1, pp. 49–69, 2011.

[64] A. Leonidis, D. Arampatzis, N. Louloudakis, and C. Stephanidis, “The AmI-Solertis System: Creating User Experiences in Smart Environments,” 2017.

[65] M. Turner, D. Budgen, and P. Brereton, “Turning software into a service,” *Computer*, vol. 36, no. 10, pp. 38–44, 2003.

[66] S. Newman, *Building microservices: designing fine-grained systems*. O’Reilly Media, Inc., 2015.

[67] H. Chen, F. Perich, T. Finin, and A. Joshi, “Soupa: Standard ontology for ubiquitous and pervasive applications,” in *The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004.*, 2004, pp. 258–267.

[68] A. Leonidis, M. Korozi, G. Margetis, D. Grammenos, and C. Stephanidis, “An intelligent hotel room,” in *International Joint Conference on Ambient Intelligence*, 2013, pp. 241–246.

[69] M. Batty *et al.*, “Smart cities of the future,” *Eur. Phys. J. Spec. Top.*, vol. 214, no. 1, pp. 481–518, 2012.

[70] C. Larman, “Design patterns elements of reusable object-oriented software,” 2005.

[71] B. M. Michelson, “Event-driven architecture overview,” *Patricia Seybold Group*, vol. 2, no. 12, pp. 10–1571, 2006.

[72] A. Van Kesteren and D. Jackson, “The xmlhttprequest object,” *World Wide Web Consort. Work. Draft WD-XMLHttpRequest-20070618*, vol. 72, 2007.

[73] R. T. Fielding and R. N. Taylor, *Architectural styles and the design of network-based software architectures*, vol. 7. University of California, Irvine Irvine, USA, 2000.

[74] *Swagger (OpenAPI) Specification*. .

[75] A. Leonidis, G. Margetis, M. Antona, and C. Stephanidis, “ClassMATE: enabling ambient intelligence in the classroom,” *World Acad. Sci. Eng. Technol.*, vol. 66, pp. 594–598, 2010.

[76] M. R. W. Dawson, *Minds and Machines: Connectionism and Psychological Modeling*. John Wiley & Sons, 2008.

[77] P. Remagnino and G. L. Foresti, “Ambient intelligence: A new multidisciplinary paradigm,” *IEEE Trans. Syst. Man Cybern.-Part Syst. Hum.*, vol. 35, no. 1, pp. 1–6, 2005.

[78] B. Ur, E. McManus, M. Pak Yong Ho, and M. L. Littman, “Practical trigger-action programming in the smart home,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2014, pp. 803–812.

[79] B. Ur *et al.*, “Trigger-action programming in the wild: An analysis of 200,000 ifttt recipes,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, 2016, pp. 3227–3231.

[80] M. Korozi, M. Antona, A. Ntagianta, A. Leonidis, and C. Stephanidis, “LECTORSTUDIO: CREATING INATTENTION ALARMS AND INTERVENTIONS TO REENGAGE THE STUDENTS IN THE EDUCATIONAL PROCESS,” in *ICERI2017 Proceedings*, Seville, Spain, Nov. 2017, pp. 4486–4495, doi: 10.21125/iceri.2017.1212.

[81] “Wireless and smart lighting by Philips | Meet Hue,” *Philips*, Jan. 19, 2018. http://www2.meethue.com/en-us (accessed Jan. 19, 2018).

[82] C. S. GMT +01:00 CH-1227 Geneva-www cross-systems com-info@cross-systems com-Tel +41 (0)22 308 4860-Timezone:, “Motorized Blinds, Shades, Awnings and Curtains with Somfy,” *Somfy -Electric motors, remote controls and automation for rolling shutters, awnings, patio blinds, indoor blinds, shades and curtains.* https://www.somfysystems.com (accessed Nov. 22, 2018).

[83] “Amazon Alexa.” https://developer.amazon.com/alexa (accessed Nov. 22, 2018).

[84] H. Merz, T. Hansemann, and C. Hübner, *Building Automation: Communication Systems with EIB/KNX, LON and BACnet*. Springer Science & Business Media, 2009.

[85] M. Antona *et al.*, *Ambient Intelligence in the classroom: an augmented school desk*. na, 2010.

[86] “Window manager,” *Wikipedia*. Apr. 08, 2020, Accessed: May 02, 2020. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Window_manager&oldid=949745691.

[87] J. Müller, F. Alt, D. Michelis, and A. Schmidt, “Requirements and design space for interactive public displays,” in *Proceedings of the 18th ACM international conference on Multimedia*, 2010, pp. 1285–1294.

[88] B. G. Witmer and M. J. Singer, “Measuring presence in virtual environments: A presence questionnaire,” *Presence*, vol. 7, no. 3, pp. 225–240, 1998.

[89] P. Chen, X. Liu, W. Cheng, and R. Huang, “A review of using Augmented Reality in Education from 2011 to 2016,” in *Innovations in Smart Learning*, Singapore, 2017, pp. 13–18, doi: 10.1007/978-981-10-2419-1_2.

[90] M. Akçayır and G. Akçayır, “Advantages and challenges associated with augmented reality for education: A systematic review of the literature,”

Educ. Res. Rev., vol. 20, pp. 1–11, Feb. 2017, doi: 10.1016/j.edurev.2016.11.002.

[91] "AWW App | Online Whiteboard for Realtime Visual Collaboration." https://awwapp.com/ (accessed Jun. 17, 2020).

[92] "Desmos | Scientific Calculator." https://www.desmos.com/scientific (accessed Jun. 17, 2020).

[93] "Google Maps," *Google Maps*. https://www.google.com/maps/@35.3141599,25.0821839,15z (accessed Jun. 17, 2020).

[94] B. M. J. Leiner, "A genetic window-placement algorithm," p. 15.

[95] "The Sawfish WindowManager." https://sawfish.tuxfamily.org/ (accessed Apr. 30, 2020).

[96] "Bin packing problem," *Wikipedia*. Apr. 28, 2020, Accessed: Apr. 30, 2020. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Bin_packing_problem&oldid=953699017.

[97] S. Martello and P. Toth, "Lower bounds and reduction procedures for the bin packing problem," *Discrete Appl. Math.*, vol. 28, no. 1, pp. 59–70, Jul. 1990, doi: 10.1016/0166-218X(90)90094-S.

[98] R. E. Korf, "A New Algorithm for Optimal Bin Packing," p. 6.

[99] J. O. Berkey and P. Y. Wang, "Two-Dimensional Finite Bin-Packing Algorithms," *J. Oper. Res. Soc.*, vol. 38, no. 5, pp. 423–429, May 1987, doi: 10.1057/jors.1987.70.

[100] D. S. Johnson, "Fast algorithms for bin packing," *J. Comput. Syst. Sci.*, vol. 8, no. 3, pp. 272–314, Jun. 1974, doi: 10.1016/S0022-0000(74)80026-7.

[101] R. Lewis, "A general-purpose hill-climbing method for order independent minimum grouping problems: A case study in graph colouring and bin packing," *Comput. Oper. Res.*, vol. 36, no. 7, pp. 2295–2310, Jul. 2009, doi: 10.1016/j.cor.2008.09.004.

[102] D. S. Liu, K. C. Tan, S. Y. Huang, C. K. Goh, and W. K. Ho, "On solving multiobjective bin packing problems using evolutionary particle swarm

optimization," *Eur. J. Oper. Res.*, vol. 190, no. 2, pp. 357–382, Oct. 2008, doi: 10.1016/j.ejor.2007.06.032.

[103] R. Hoberg and T. Rothvoss, "A logarithmic additive integrality gap for bin packing," in *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2017, pp. 2616–2625.

[104] E. Falkenauer, *Genetic algorithms and grouping problems*. John Wiley & Sons, Inc., 1998.

[105] A. Lodi, S. Martello, and D. Vigo, "Recent advances on two-dimensional bin packing problems," *Discrete Appl. Math.*, vol. 123, no. 1–3, pp. 379–396, 2002.

[106] D. Simchi-Levi, "New worst-case results for the bin-packing problem," *Nav. Res. Logist. NRL*, vol. 41, no. 4, pp. 579–585, 1994.

[107] "Hammer.JS - Hammer.js." https://hammerjs.github.io/ (accessed Apr. 13, 2020).

[108] "npm | build amazing things." https://www.npmjs.com/ (accessed Apr. 13, 2020).

[109] "Handling Pan Gestures | Apple Developer Documentation." https://developer.apple.com/documentation/uikit/touches_presses_and_gestures/handling_uikit_gestures/handling_pan_gestures (accessed Apr. 13, 2020).

[110] "Handling Long-Press Gestures | Apple Developer Documentation." https://developer.apple.com/documentation/uikit/touches_presses_and_gestures/handling_uikit_gestures/handling_long-press_gestures (accessed Apr. 13, 2020).

[111] "Handling Rotation Gestures | Apple Developer Documentation." https://developer.apple.com/documentation/uikit/touches_presses_and_gestures/handling_uikit_gestures/handling_rotation_gestures (accessed Apr. 13, 2020).

[112] "Handling Pinch Gestures | Apple Developer Documentation." https://developer.apple.com/documentation/uikit/touches_presses_and_

gestures/handling_uikit_gestures/handling_pinch_gestures (accessed Apr. 13, 2020).

[113] "Handling Swipe Gestures | Apple Developer Documentation." https://developer.apple.com/documentation/uikit/touches_presses_and_gestures/handling_uikit_gestures/handling_swipe_gestures (accessed Apr. 13, 2020).

[114] "Handling Tap Gestures | Apple Developer Documentation." https://developer.apple.com/documentation/uikit/touches_presses_and_gestures/handling_uikit_gestures/handling_tap_gestures (accessed Apr. 13, 2020).

[115] "MyScript." https://www.myscript.com (accessed Apr. 17, 2020).

[116] "Angular." https://angular.io/ (accessed May 07, 2020).

[117] "angular-gridster2," *npm*. https://www.npmjs.com/package/angular-gridster2 (accessed May 07, 2020).

[118] "tiberiuzuld/angular-gridster2," *GitHub*. https://github.com/tiberiuzuld/angular-gridster2 (accessed May 07, 2020).

[119] "JavaScript Event Propagation - Tutorial Republic." https://www.tutorialrepublic.com/javascript-tutorial/javascript-event-propagation.php (accessed May 08, 2020).

[120] "Redis." https://redis.io/ (accessed May 20, 2020).

[121] J. Rieman, M. Franzke, and D. Redmiles, "Usability evaluation with the cognitive walkthrough," in *Conference companion on Human factors in computing systems*, 1995, pp. 387–388.

[122] J. Nielsen, *Usability engineering*. Morgan Kaufmann, 1994.

[123] A. S. for P. Affairs, "System Usability Scale (SUS)," Sep. 06, 2013. /how-to-and-tools/methods/system-usability-scale.html (accessed May 20, 2020).

[124] C. Katsanos, N. Tselios, and M. Xenos, "Perceived usability evaluation of learning management systems: a first step towards standardization of the System Usability Scale in Greek," in *2012 16th Panhellenic Conference on Informatics*, 2012, pp. 302–307.

[125] W. L. in R.-B. U. Experience, “Thinking Aloud: The #1 Usability Tool,” *Nielsen Norman Group*. https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/ (accessed Apr. 05, 2020).

[126] J. Brooke, “SUS: a retrospective,” *J. Usability Stud.*, vol. 8, no. 2, pp. 29–40, 2013.

[127] “MeasuringU: Measuring Usability with the System Usability Scale (SUS).” https://measuringu.com/sus/ (accessed May 29, 2020).

Appendix A

User-Based Evaluation Task Scenarios

Introduction

You teach Human Computer Interaction at the Computer Science Department of University of Crete. The department features an Intelligent Classroom that has been assigned to your course. For today's lecture, you want to introduce the Process of **Design Thinking**.

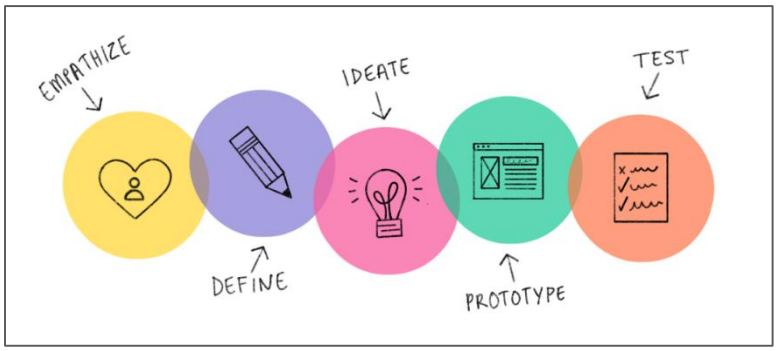


Figure 62: Design Thinking Process.

Before the lecture, you use your PC at your university office to prepare the contents of the board. You launch the "CognitOS Classboard Desktop" application and load a presentation file, two interesting images and one video. The first picture (Figure 63) depicts a technique for empathic modeling and the second an example of prototyping (Figure 64).

Figure 63: Empathic modelling.

Figure 64: Prototyping.

Task 1 – A

You enter the classroom and the board has already been initialized with the content you loaded previously. Before you start the lecture, you want to remind to the students the upcoming deadline of the project. So…

- In the main wall, you launch the application that shows the course schedule.
- You find the deadline of project's next phase and inform the students.
- In the next step, you move the application on the side wall.

Task 1 – B

On the side wall, there is an invisible workspace where you have placed the application that shows the students deliverables.

- Aiming to see which teams have already submitted their project, display the corresponding workspace.

Task 2

You start the lecture based on the slides of the presentation file.

- You reach slide 4, where is an image that depicts the process of Design Thinking (Figure 62).

- In order to make the lesson more interactive, you take a Capture of the image from the ppt.

Task 3 – A

- **Enlarge** and **Move** the snapshot relatively high on the board. On the right side of the presentation file.
- … and select not to be able to move.

Task 3 – B

- Move the two images you have uploaded before the lecture near to the snapshot (in appropriate positions as visualized in Figure 65).
- … and select not to be able to move.

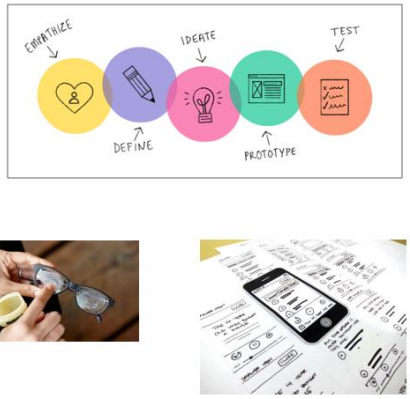


Figure 65: Board's state after the completion of Task 3.

Task 4

You want to show that the first image (Figure 63) is corresponding to the first step of Design Thinking.

- Draw an arrow (annotate) from the Empathize step to the appropriate image.

After the explanation of the Empathize step…

- Find in the second board the video that you have uploaded before the lecture.

Task 5

- Maximize the video application.
- Start the video.

When the video is completed…

- Return to the first board (which contains the presentation file).

Task 6

Continue by explaining the next steps of Design Thinking, when you reach the Prototype step…

- Draw an arrow (annotate) from the Prototype step to the appropriate image (Figure 64).

After the explanation of the Prototype step…

- You want to show to the students a quick example of creating a prototype, so you create a 3rd board.

Task 7

- Launch the application that allows you to write as if you were writing on a whiteboard.
- Design a login box (as visualized in Figure 66)

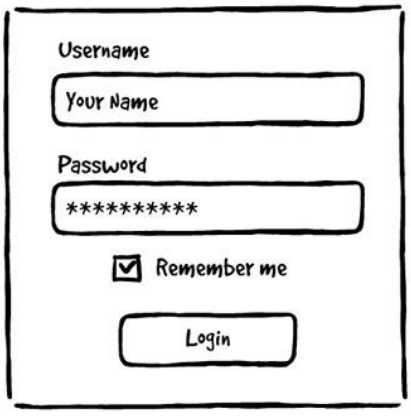


Figure 66: Login box prototype.

Task 8

- Return to the first board (which contains the presentation file).
- Take a capture of the three images with the arrows you have previously drawn (Figure 67).
- And send the capture to the student's.

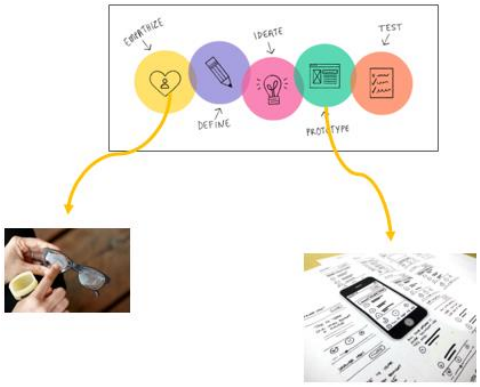


Figure 67: The captured area.

Task 9 – A

Once you have completed the lecture you decide to give them a mini quiz. You verbally describe them the question and give them 10 minutes to answer.

- To keep track of passing time, launch the stopwatch application and start the timer.

Task 9 – B

In general, you move during quizzes to avoid/ catch students cheating.

- Set the stopwatch to follow you so you can press the stop button when the time comes.

After a while…

- Stop the stopwatch.
- Set the stopwatch application to stop following you.

Task 10

When the quiz is over, inform the students that they can leave the classroom and…

- Performing a **"C"** gesture with the pen, switch off the board and leave the classroom.