

# Software Engineering 2022W

## Introduction



**Wesley Klewerton**  
**Guez Assunção**  
343.001, 343.302

**Cosmina-Cristina**  
**Ratiu**  
343.303

**Luciano Marchezan**  
**de Paula**  
343.309

# Schedule

Topic	Date	Lecturer
E1: Familiarize with Dronology	5-Oct-22	Wesley
E2: Use Case Analysis	12-Oct-22	Luciano
E3: Tracing Use Cases to EARS Requirements	19-Oct-22	Cosmina
E4: Requirements Engineering Mission Planning	9-Nov-22	Wesley/Cosmina
E5: Define Architecture and Design	16-Nov-22	Luciano
E6: Define Mission Planning Extension	30-Nov-22	Cosmina
E7: Mission Planning Cost Estimation	7-Dec-22	Wesley
E8: Quality Assurance	14-Dec-22	Luciano
E9: Test Case Design	11-Jan-23	Wesley
E10: In-class exercise	18-Jan-23	Wesley

# Individual and Team Exercises

- There are individual and team exercises
  - Make sure to describe what is done by each team member
- Form groups with 5 members
  - Use the Moodle Grouptool

 Teams for the exercises

Topic	Date	Type
E1: Familiarize with Dronology	5-Oct-22	Individual
E2: Use Case Analysis	12-Oct-22	Individual
<u>E3: Tracing Use Cases to EARS Requirements</u>	<u>19-Oct-22</u>	<u>Team</u>
E4: Requirements Engineering Mission Planning	9-Nov-22	Individual
<u>E5: Define Architecture and Design</u>	<u>16-Nov-22</u>	<u>Team</u>
E6: Define Mission Planning Extension	30-Nov-22	Individual
E7: Mission Planning Cost Estimation	7-Dec-22	Individual
<u>E8: Quality Assurance</u>	<u>14-Dec-22</u>	<u>Team</u>
E9: Test Case Design	11-Jan-23	Individual
E10: In-class exercise	18-Jan-23	Individual

# Cumulative Exercises

- Each next exercise will be based the output of the previous one
- A task accurately done will ease the following tasks

Topic	Date	Type
E1: Familiarize with Dronology	5-Oct-22	Individual
E2: Use Case Analysis	12-Oct-22	Individual
E3: Tracing Use Cases to EARS Requirements	19-Oct-22	Team
E4: Requirements Engineering Mission Planning	9-Nov-22	Individual
E5: Define Architecture and Design	16-Nov-22	Team
E6: Define Mission Planning Extension	30-Nov-22	Individual
E7: Mission Planning Cost Estimation	7-Dec-22	Individual
E8: Quality Assurance	14-Dec-22	Team
E9: Test Case Design	11-Jan-23	Individual
E10: In-class exercise	18-Jan-23	Individual

# Grades

- Each exercise will be graded between 0 and 10
  - The final grade will be the sum of all (10) exercises
- Grades will be given individually
  - Even in team exercises
- Empty/corrupted files delivered
  - Grade with 0
- Plagiarism/Copies
  - Grade with 0
  - Whole team fails!

Final Grade	Total
1 (SGT)	$\geq 87.5$
2 (GUT)	$\geq 75$ and $< 87.5$
3 (BEF)	$\geq 62.5$ and $< 75$
4 (GEN)	$\geq 50$ and $< 62.5$
5 (NGD)	$< 50$

# Software Engineering 2022W

## E1: Familiarize with Dronology



Wesley Klewerton  
Guez Assunção  
343.001, 343.302

Cosmina-Cristina  
Ratiu  
343.303

Luciano Marchezan  
de Paula  
343.309

# Schedule

Topic	Date	Lecturer
<b>E1: Familiarize with Dronology</b>	<b>5-Oct-22</b>	<b>Wesley</b>
E2: Use Case Analysis	12-Oct-22	Luciano
E3: Tracing Use Cases to EARS Requirements	19-Oct-22	Cosmina
E4: Requirements Engineering Mission Planning	9-Nov-22	Wesley/Cosmina
E5: Define Architecture and Design	16-Nov-22	Luciano
E6: Define Mission Planning Extension	30-Nov-22	Cosmina
E7: Mission Planning Cost Estimation	7-Dec-22	Wesley
E8: Quality Assurance	14-Dec-22	Luciano
E9: Test Case Design	11-Jan-23	Wesley
E10: In-class exercise	18-Jan-23	Wesley

# Introduction


- Learning goal
  - Understand the system Dronology and its artifacts
- Input
  - Dronology Dataset
  - Dronology ICSE-NIER paper
  - Introductory lecture (Dr. Michael Vierhauser)
- Output
  - Questionnaire response




# Dronology

## Unmanned Aerial Systems Management and Control

Dronology is an Open-Source Unmanned Aerial System (UAS) developed at the [University of Notre Dame](#). The project has two main objectives:

 First, to **establish a research environment** for studying various aspects of software and systems engineering for cyber-physical systems – e.g., runtime monitoring, safety analysis, and product line development.

 Second, to **provide a framework for controlling and coordinating the flight** of individual UAS, formations, and swarms in order to support applications such as search-and-rescue, surveillance, and scientific data collection.

These two goals are closely related — as developing a research environment for investigating challenges in safety-critical software systems involves building a system with safety-critical implications.



# Dronology Dataset

- Use Cases (UCx)
- Detailed Use Cases (SCx)
- Requirements
- Components

	A	B	C	D	E	
1	Nr.	Type	ID	Name	Description	
2	1	Cmp	CO-90	GCS Middleware	Handles connections between Dronology and Ground Control Stations (GCS). Forwards commands monitoring and other messages from Dronology to its registered GCS and passes messages describing the state of the UAVs managed by each GCS back to Dronology.	DD-354 - DD-361 - DD-720 - DD-721 - DD-734 - DD-735 - DD-736
3	2	Cmp	CO-91	GCS Middleware	Python based system that manages and controls UAVs. Communicates with Dronology via the Ground Station middleware. Each GCS is responsible for communicating directly with each UAV sending it commands and monitoring its state including its current position flight mode and health.	DD-740 - DD-742 - DD-753 - DD-755 - DD-756
4	3	Cmp	CO-105	UI Real-Time Flight View	Manages all aspects of displaying flights and UAVs in real-time and interacting with them. The flight view displays active routes UAV coordinates and their current health. The map uses zoom and panning features to follow one or more selected UAV.	DD-113 - DD-121 - DD-690 - DD-692 - DD-691 - RE-693 - RE-694
5	4	Cmp	CO-184	Internal Simulator	The internal simulator provides low-fidelity features for supporting quick initial tests of a virtual UAV. Features include takeoff goto land and battery health.	RE-593 - RE-594 - RE-595
6	5	Cmp	CO-472	Flight Route Manager	Provides capabilities for creating and managing flight routes within Dronology.	DD-502 - DD-504 - DD-505
7	6	Cmp	CO-473	Coordinate System	Provides 3 dimensional coordinate system currently based on longitude latitude and altitude. Provides conversion between various coordinate systems.	DD-177 - DD-23 - DD-510 - RE-516 - RE-517
8	7	Cmp	CO-474	Flight Manager	Provides basic flight related capabilities for UAVs.	DD-520 - DD-522 - DD-523
9	8	Cmp	CO-475	Flight Scheduling and Execution	Provides capabilities for executing flights assigned to a UAV.	DD-532 - DD-533 - DD-534
10	9	Cmp	CO-476	Mission Planning	Provides capabilities for executing missions with multiple UAVs	DD-362 - DD-468 - DD-546 - RE-541 - RE-542
11	10	Cmp	CO-477	Single UAV Flight Plan Scheduler	Single UAV flight plan assignments (taking an existing route instantiating it and assigning to a UAV)	DD-27 - DD-548 - DD-561 - DD-81 - DD-RE-551 - RE-555 - RE-556
12	11	Cmp	CO-480	Flight Patterns	Provides capabilities for creating and executing complex flight patterns for multiple UAVs	RE-562 - RE-563
13	12	Cmp	CO-481	Object Avoidance	Collision Avoidance API and general services	DD-564 - DD-565 - DD-566

+ 
≡ 
Component 
Requirement 
Design Definition 
Sub-Tasks 
Hazards 
UC1 
UC2 
UC3 
> 
Explorator

# ICSE NIER Paper

2021 IEEE/ACM 43rd International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)

- Overview of Dronology

## Towards a Model-Integrated Runtime Monitoring Infrastructure for Cyber-Physical Systems

Michael Vierhauser, Hussein Marah,  
Antonio Garmendia  
*Johannes Kepler University Linz*  
michael.vierhauser@jku.at

Jane Cleland-Huang  
*University of Notre Dame*  
South Bend, IN, USA  
janeclelandhuang@nd.edu

Manuel Wimmer  
*CDL-MINT, Johannes Kepler University Linz*  
Linz, Austria  
manuel.wimmer@jku.at

**Abstract**—Runtime monitoring is essential for ensuring the safe operation and enabling self-adaptive behavior of Cyber-Physical Systems (CPS). It requires the creation of system monitors, instrumentation for data collection, and the definition of constraints. All of these aspects need to evolve to accommodate changes in the system. However, most existing approaches lack support for the automated generation and setup of monitors and constraints for diverse technologies and do not provide adequate support for evolving the monitoring infrastructure. Without this support, constraints and monitors can become stale and become less effective in long-running, rapidly changing CPS. In this “new and emerging results” paper we propose a novel framework for model-integrated runtime monitoring. We combine model-driven techniques and runtime monitoring to automatically generate large parts of the monitoring framework and to reduce the maintenance effort necessary when parts of the monitored system

monitoring environment synchronized with the evolving system can be a tedious and time-consuming endeavor; however, neglecting to do so can lead to decreasing effectiveness as constraints start producing false-positive errors and other important constraints are missing.

The Model-driven Engineering (MDE) paradigm [8], provides methods for addressing evolution and maintenance issues related to software systems. In such a scenario, changes are performed solely to the model and system code is generated automatically, without the need to manually adapt and modify different parts of the system. In this sense, it seems promising to employ the same technique to model a monitoring framework, define events, data that needs to be collected, and link

# Questionnaire

- Considering the artifacts (dataset and paper) of Dronology, answer the following questions:
  - a. What is Dronology about?
  - b. What are the main features of it?
  - c. What are the stakeholders of Dronology? (please consider the Stakeholder Onion Diagram)
  - d. What are the available artifacts and what are they used for?

# Hand-in procedure and Grading Criteria

- Hand-in procedure
  - Answers should be added directly in the Moodle task.
- Grading Criteria
  - Correctness (correct info) of the answers
  - Completeness (no missing info) of the answers
  - Each of the 4 questions: 2.5 points