

# Cerberus: Automated Synthesis of Enforcement Mechanisms for Security-sensitive Business Processes

Luca Compagna<sup>2</sup>, **Daniel Ricardo dos Santos**<sup>1,2,3</sup>,  
Serena Elisa Ponta<sup>2</sup>, Silvio Ranise<sup>1</sup>

<sup>1</sup>Fondazione Bruno Kessler (FBK)

<sup>2</sup>SAP Labs France

<sup>3</sup>University of Trento



## SECENTIS

A European Industrial Doctorate on Security and Trust

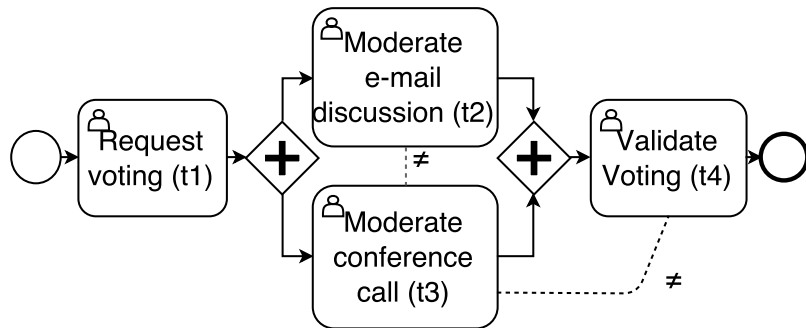
# Outline

- 1 Introduction
- 2 Using Cerberus - demo
- 3 Conclusion

# Context

- A security-sensitive business process is a structured collection of **tasks** with:
  - **Authorization policy**: which users are entitled to execute which tasks
  - **Authorization constraints**: e.g., some tasks must be performed by the same/different users
- Policy and constraints are crucial to **comply with regulations** and prevent frauds, but business continuity must not be endangered:
  - It must be possible to **complete** the process while satisfying the policy and constraints.

# Example



## Authorization policy

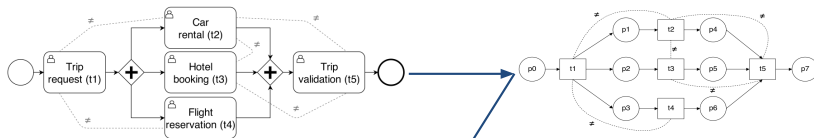
t1	A, B, C
t2	A, B, C
t3	A, B
t4	A

**Satisfying runs:** t1(C),t2(A),t3(B),t4(A) or  
t1(A), t3(B), t2(C), t4(A)

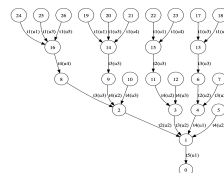
**Avoid:** t1(A), t2(B), ...

# Solution

- Cerberus synthesizes, at **design-time**, monitors capable of answering, at **run-time**, user requests to execute tasks
  - request is granted if user is **authorized** (policy), no **constraint** is violated and the execution can still **terminate** (there are users who can perform the next tasks)
- Synthesized monitors are **parametric** in the authorization **policy**
  - can be combined at run-time with authorization policies dedicated to different instances of the process.



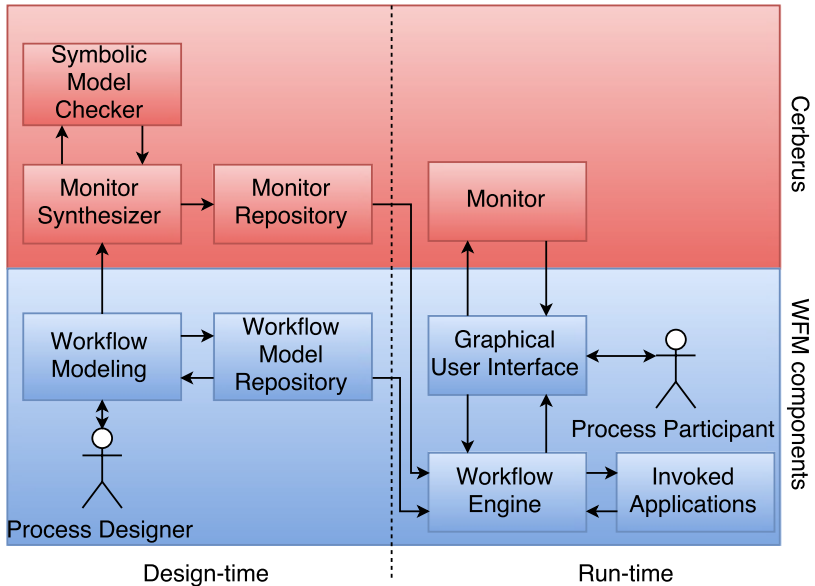
event	enabled		action	
	CF	Auth	CF	Auth
$t1(u)$	$p0 \wedge \neg d_{t1}$	$a_{t1}(u)$	$p0, p1, p2, p3, d_{t1} := F, T, T, T, T$	$h_{t1}(u) := T$
$t2(u)$	$p1 \wedge \neg d_{t2}$	$a_{t2}(u) \wedge \neg h_{t3}(u) \wedge \neg h_{t1}(u)$	$p1, p4, d_{t2} := F, T, T$	$h_{t2}(u) := T$
$t3(u)$	$p2 \wedge \neg d_{t3}$	$a_{t3}(u) \wedge \neg h_{t2}(u)$	$p2, p5, d_{t3} := F, T, T$	$h_{t3}(u) := T$
$t4(u)$	$p3 \wedge \neg d_{t4}$	$a_{t4}(u) \wedge \neg h_{t1}(u)$	$p3, p6, d_{t4} := F, T, T$	$h_{t4}(u) := T$
$t5(u)$	$p4 \wedge p5 \wedge p6 \wedge \neg d_{t5}$	$a_{t5}(u) \wedge \neg h_{t3}(u) \wedge \neg h_{t2}(u)$	$p4, p5, p6, p7, d_{t5} := F, F, F, T, T$	$h_{t5}(u) := T$



#	Token in	Auth					$can\_do(u, t)$	Resp.
		$h_{t1}$	$h_{t2}$	$h_{t3}$	$h_{t4}$	$h_{t5}$		
0	$p0$	-	-	-	-	-	$(a, t1)$	deny
1	$p0$	-	-	-	-	-	$(b, t1)$	grant
2	$p1, p2, p3$	$b$	-	-	-	-	$(b, t2)$	deny
3	$p1, p2, p3$	$b$	-	-	-	-	$(a, t2)$	grant
4	$p4, p2, p3$	$b$	$a$	-	-	-	$(c, t3)$	grant
5	$p4, p5, p3$	$b$	$a$	$c$	-	-	$(a, t4)$	grant
6	$p4, p5, p6$	$b$	$a$	$c$	$a$	-	$(b, t5)$	grant
7	$p7$	$b$	$a$	$c$	$a$	$b$	-	-

# Integration

- Cerberus can be integrated in many **workflow management systems**
  - It is transparent to process designers, and does not require any knowledge beyond usual BP modeling
  - Monitors can be synthesized in Datalog and SQL
- We integrated it into **SAP HANA Operational Intelligence**
  - BPMN modeling IDE based on eclipse
  - Enactment platform on top of an in-memory database (BPMN models translated to SQL)





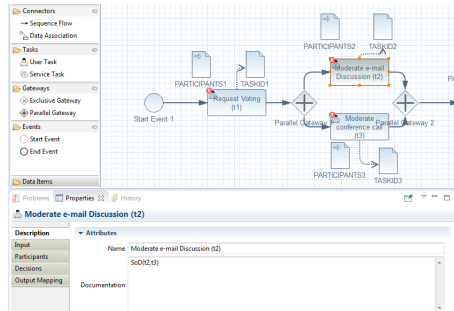
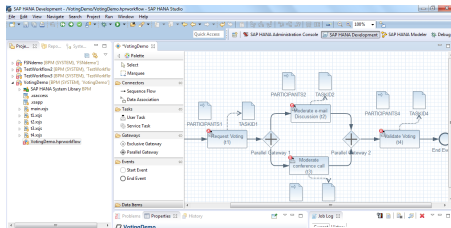
# Outline

- 1 Introduction
- 2 Using Cerberus - demo
- 3 Conclusion

Three steps:

- Design-time
- Deployment (monitor synthesis)
- Run-time

## Design-time: BPMN modeling and constraint specification



## Monitor synthesis:

The screenshot shows the SAP HANA Studio interface. On the left, the Project Explorer displays a project named 'VotingDemo'. The 'Generate' menu option is highlighted. The main editor shows the JavaScript code for 'main.xsjs', which defines a workflow API and its execution context.

```

main.xsjs
var workflowAPI = $.import("VotingDemo.VotingDemo.v1.api", "PublicAPI");
var processStartParams = {
  PARTICIPANTS1: [{PRINCIPAL_ID: "USERA"}, {PRINCIPAL_ID: "USERB"}, {PRINCIPAL_ID: "USERC"}],
  PARTICIPANTS2: [{PRINCIPAL_ID: "USERA"}, {PRINCIPAL_ID: "USERB"}, {PRINCIPAL_ID: "USERC"}],
  PARTICIPANTS3: [{PRINCIPAL_ID: "USERA"}, {PRINCIPAL_ID: "USERB"}],
  PARTICIPANTS4: [{PRINCIPAL_ID: "USERA"}],
};

try {
  var conn = $.db.getConnection();
  var executionContext = new workflowAPI.ExecutionContext({
    connection: conn
  });

  var workflowInstance = workflowAPI.startWorkflow({
    data: processStartParams,
    executionContext: executionContext
  });

  conn.commit();

  var id = workflowInstance.getId().toString();
  var isRunning = workflowInstance.isRunning();
  var result = workflowInstance.getResult();

  var response = "<html>"
  response += "<table border='1'>"
  response += "<tr>"
  response += "<td>Id:</td>"
  response += "<td>" + id + "</td>"
  response += "<tr>"
  response += "<td>IsRunning:</td>"
  response += "<td>" + isRunning + "</td>"
  response += "<tr>"
  response += "<td>Result:</td>"
  response += "<td>" + result + "</td>"
  response += "</tr>"
  response += "</table>"
  response += "</html>"
} catch (e) {
  response = "<html>"
  response += "<table border='1'>"
  response += "<tr>"
  response += "<td>Error:</td>"
  response += "<td>" + e.getMessage() + "</td>"
  response += "</tr>"
  response += "</table>"
  response += "</html>"
}

return response;

```

Below the code editor, the 'Properties' tab is active, showing the properties of the 'VotingDemo.hprworkflow - VotingDemo' workflow:

Property	Value
Info	
derived	false
editable	true

# Stored monitor:

## VotingDemo\_VotingDemo

### Details

```

131
132 --monitor auhtorization check for UserTask_2
133 ELSEIF TASK_ID_IN = m_task_id_UserTask_2 THEN
134   var_out =
135     SELECT
136       CASE WHEN T.STATUS='READY' AND (A.ROLE_TYPE='PARTICIPANT' OR A.ROLE_TYPE='OWNER') THEN 1 ELSE 0 END AS "CAN_COMPLETE",
137       T.TASK_ID
138     FROM "TASKMGT"."sap.bc.taskmgt.task::TASK" AS T INNER JOIN "TASKMGT"."sap.bc.taskmgt.task::ASSIGNMENT" AS A ON T.TASK_ID = A.TASK_ID
139     WHERE (T.CATEGORY = 'TASK' OR T.CATEGORY = 'CHECKLIST')
140     AND A.PRINCIPAL_ID IN
141     (
142       --monitor query
143       SELECT DISTINCT Z1."USER_NAME" FROM "SYS"."USERS" AS Z1 WHERE doneUserTask_2 = 0 AND doneUserTask_4 = 0 AND doneUserTask_1 = 1 AND d
144       UNION
145       SELECT DISTINCT Z2."USER_NAME" FROM "SYS"."USERS" AS Z1, "SYS"."USERS" AS Z2 WHERE doneUserTask_2 = 0 AND doneUserTask_4 = 0 AND don
146       UNION
147       SELECT DISTINCT Z1."USER_NAME" FROM "SYS"."USERS" AS Z1, "SYS"."USERS" AS Z2 WHERE doneUserTask_2 = 0 AND doneUserTask_3 = 0 AND don
148       UNION
149       SELECT DISTINCT Z3."USER_NAME" FROM "SYS"."USERS" AS Z1, "SYS"."USERS" AS Z2, "SYS"."USERS" AS Z3 WHERE doneUserTask_2 = 0 AND doneU
150       UNION
151       SELECT DISTINCT Z2."USER_NAME" FROM "SYS"."USERS" AS Z1, "SYS"."USERS" AS Z2 WHERE doneUserTask_2 = 0 AND doneUserTask_4 = 0 AND don
152       UNION
153       SELECT DISTINCT Z1."USER_NAME" FROM "SYS"."USERS" AS Z1 WHERE doneUserTask_2 = 0 AND doneUserTask_4 = 0 AND doneUserTask_1 = 1 AND d
154       --end of monitor query
155     ) AND A.PRINCIPAL_ID = SESSION_USER;
156
157 --monitor auhtorization check for UserTask_3
158 ELSEIF TASK_ID_IN = m_task_id_UserTask_3 THEN
159   var_out =
160     SELECT
161       CASE WHEN T.STATUS='READY' AND (A.ROLE_TYPE='PARTICIPANT' OR A.ROLE_TYPE='OWNER') THEN 1 ELSE 0 END AS "CAN_COMPLETE",
162       T.TASK_ID
163     FROM "TASKMGT"."sap.bc.taskmgt.task::TASK" AS T INNER JOIN "TASKMGT"."sap.bc.taskmgt.task::ASSIGNMENT" AS A ON T.TASK_ID = A.TASK_ID
164     WHERE (T.CATEGORY = 'TASK' OR T.CATEGORY = 'CHECKLIST')
165     AND A.PRINCIPAL_ID IN
166     (
167       --monitor query
168       SELECT DISTINCT Z2."USER_NAME" FROM "SYS"."USERS" AS Z1, "SYS"."USERS" AS Z2 WHERE doneUserTask_3 = 0 AND doneUserTask_4 = 0 AND don

```

## Run-time:

The first screenshot shows the 'My Workbox' for user 'USERA'. It displays a table with one task:

Type	Subject	Due Date	Status	Priority
📅	t1 - VotingDemo		Ready	Medium

The second screenshot shows the task 't1 - VotingDemo' for user 'USERA' as 'Task 130 completed'.

The third screenshot shows the task 't3 - VotingDemo' for user 'USERB' as 'Task 140 could not be completed because the user is not authorized'.

# Outline

- 1 Introduction
- 2 Using Cerberus - demo
- 3 Conclusion

# Conclusion

- We have [other papers](#) describing in more details the techniques used in the tool:
  - “Automated Synthesis of Run-time Monitors to Enforce Authorization Policies in Business Processes” in ASIACCS 2015
  - “Modularity for Security-Sensitive Workflows” in arXiv
  - “Assisting the Deployment of Security-Sensitive Workflows by Finding Execution Scenarios” in DBSec 2015
- The tool is [under development](#) and there is a pilot project for internal use in SAP (payment approval workflows)
- Not yet available for public use



Thank you!

`dossantos@fbk.eu`

`www.secentis.eu`



UNIVERSITY  
OF TRENTO