An Integrated Approach for Context-Aware Development

Aurora Macías – Indra Sistemas

Elena Navarro – University of Castilla-La Mancha

UCLM
UNIVERSIDAD DE CASTILLA-LA MANCHA

LoUISE
Laboratorio de Interacción con el Usuario e Ingeniería del Software

Contents

Introduction

Introduction

- Mobile devices omnipresent / ubiquitous systems increasingly popular
- Context-Aware systems:
  - Capable to **adapt** their operations **without explicit user interaction**
  - Great potential for **increasing quality** (i.e.: usability, effectiveness)
  - Satisfy a variety of requirements: SoC, acquisition / storage / interpretation of context, transparent communications, resource discovery
  - **Requirements** should be **supported** by **Software Architecture**

Background: Approaches to the development of Context-Aware Systems

Background

- Context Toolkit

- MAPE-K loop

- Layer Approach

Background. Context Toolkit

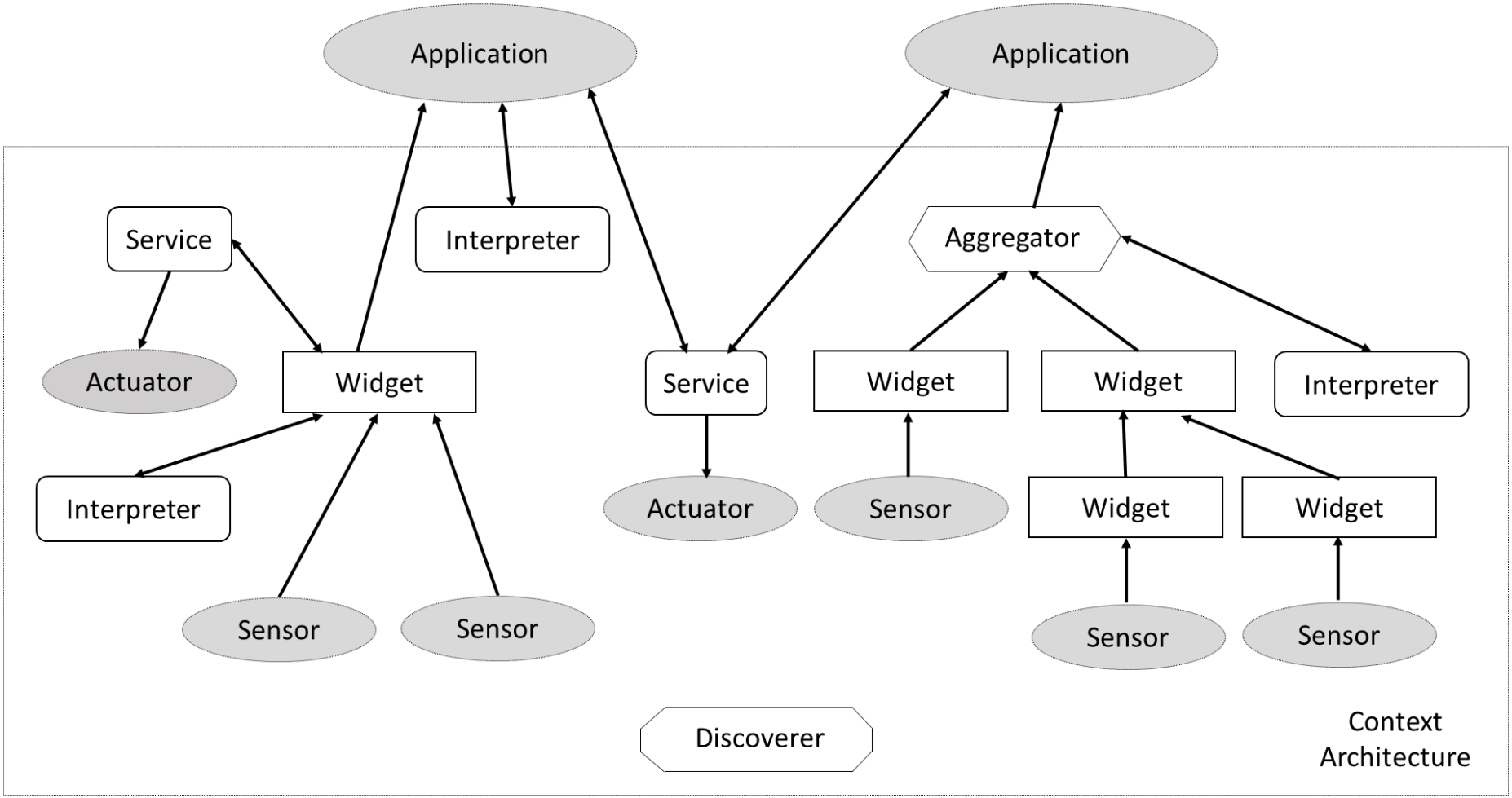
- **Different abstractions** or software components to offer SoC, facilitate reuse, and satisfy C-A requirements

- **Independently executable** and **deployable**

- **Context Architecture**: supports context information acquisition and delivery, and execution of common actions

- Application logic outside context architecture

Background. Context Toolkit

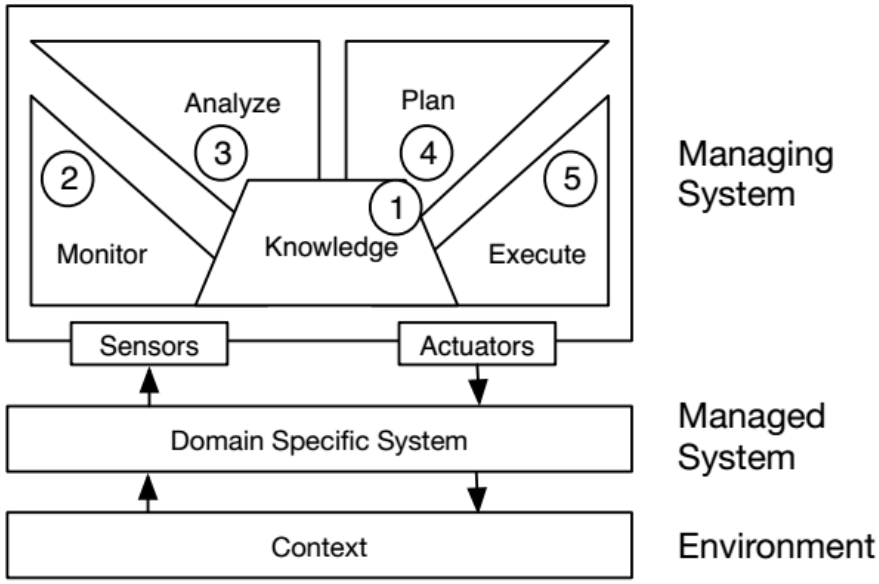


Background. MAPE-K Loop

- **Self-adaptive systems** introduced for C-A development

- **SoC key aspect of MAPE-K Loop** configuration to achieve adaptation goals

- Execution of components in cycle, **communication** 'among them' **through Knowledge**

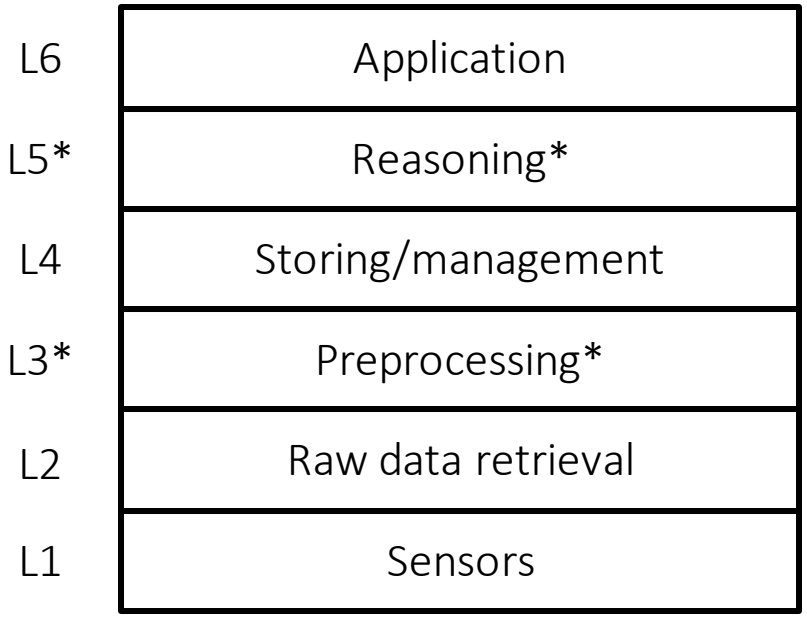


Background. Layer Approach

- Collects the aspects common to most of the context-aware architectures

- Entails the SoC needed to acquire the context and to reuse the components

- Each layer gets information from the layer underneath



Background.
Common Aspects of the Analyzed Frameworks

Responsibility	CTK	Layer Approach	MAPE-K
Acquisition of data (from sensors and other sources) as elements of simple context	Widget	Raw data retrieval	Monitor Managed system
Resolution of conflicts when obtaining contextual data from various sources	Widget	Pre-processing	Monitor
Data pre-processing (simple inference)	Interpreter	Pre-processing	Monitor
Aggregation of multiple logically related context information elements	Aggregator	Pre-processing	Monitor
Complex inference (reasoning)	Interpreter	Reasoning	Analyze
Updating of information	Widget Aggregator Discoverer	Storing/management	Monitor
Publishing information (through notification of change or methods for querying) related to relevant aspects of the system and the environment	Widget Aggregator Discoverer	Storing/management	Knowledge
Changing of status information (configuration)	Service	Storing/management	Plan Execute
Controling (of the system) or changing in the environment by using actuators	Service	Application	Managed system
Publishing (through queries) of adaptation / action objectives	---	---	Knowledge

Background.
Common Aspects of the Analyzed Frameworks

- Exposure of adaptation / action objectives not supported by CTK nor the layer approach

- Each CTK-component or layer manages its own adaptation / action information without exposing it

- **Advantages** of **non-exposure** of adaptation goals:

  - Simplified maintenance: modify and redeploy only 1 component in case of change of objectives:

    - less inactivity time (subsystem inactive, not the entire system) leads to increased availability

  - More sources for storing data:

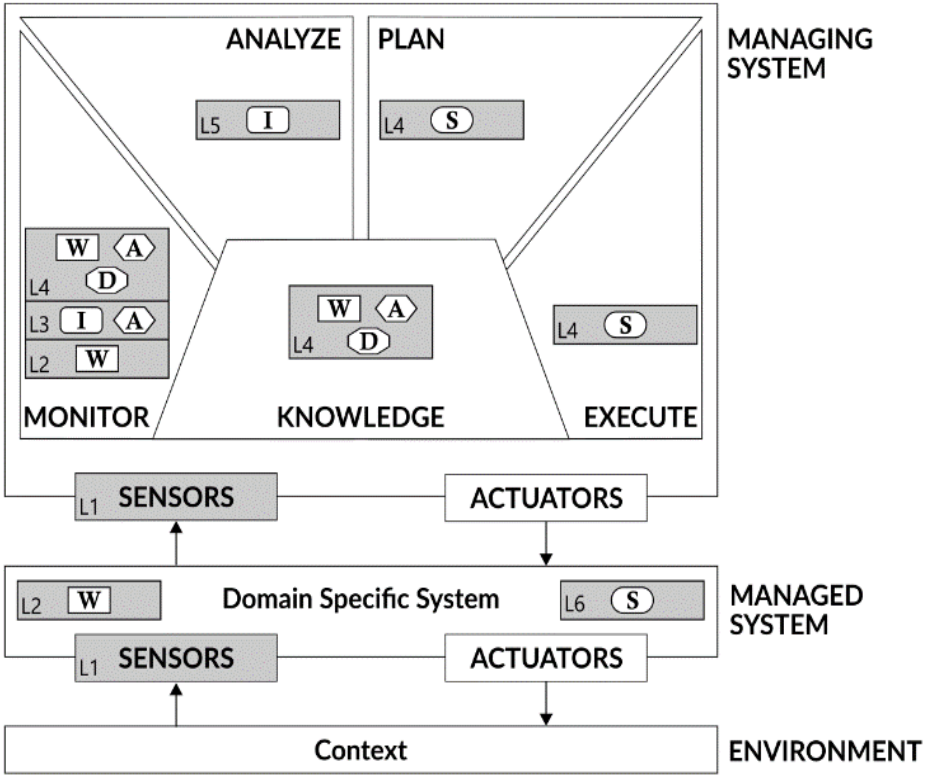
    - number of bottlenecks reduced (increased performance)

Background.
Common Aspects of the Analyzed Frameworks

Conceptual composition in terms of **responsibilities** among components of different frameworks

For instance:

- **MAPE-K Monitor** equivalent to:
  - **Raw data recovery and pre-processing layers**
  - **Part of the management / storage layer**
- **Raw data recovery layer** equivalent to **CTK context widget** (single element)
- **Pre-processing layer** equivalent **CTK interpreter** and **aggregator**



Background.

Common Aspects of the Analyzed Frameworks

- *CTK* designed to (+) satisfy C-A requirements (SoC, acquisition / storage / interpretation of context, transparent communications, resource discovery) (-) but SoC regarding inference is not properly addressed

- *Layer approach* derived to (+) consider most common responsibilities of C-A systems but (-) does not tackle explicitly resources discovery

-*MAPE-K* proposed to (+) provide the SoC needed to achieve self-adaptation goals but presents (-) different disadvantages mentioned

- Thus, all proposals should offer a **higher SoC**

An Integrated Proposal

An Integrated Proposal.
Architecture Elements

- Proposed architectural framework elements based on CTK abstractions
  - CTK provides the highest SoC and cohesion degree of the frameworks analyzed

- CTK elements concerns modified and regrouped to achieve a higher SoC

- New proposed framework elements match resulting concerns

- Service component renamed to ‘function’ to avoid possible ambiguity when considering implementation aspects

An Integrated Proposal.
Architecture Elements

Component	Responsibilities
Context Widget	(i) Acquisition, using sensors, of environment status information related to a context element which may be optionally composed. It also resolves conflicts when datum is obtained from different sources. (ii) Publishing information related to contextual elements by using (a) notification of significative changes of the context or (b) context polling methods. (iii) Publishing information related to the context acquisition by using request methods of additional attributes (sensor type, data acquisition method, sensor exactitude/precision, etc.). (iv) Registration of the context elements into the acquired context history.
Aggregator	(i) Gathering of multiple context elements related to an entity in order to facilitate access to such information. (ii) Publishing aggregated contextual information related to the corresponding entity by using (a) notification of changes of the component context or (b) context polling methods. (iii) Registration of the context information into the acquired context history.
Interpreter**	(i) **Simple inference or derivation**: transformation of context atomic information using auxiliary sources of information.
Reasoner*	(i) **Complex inference or reasoning**: gathering new context information of higher abstraction level by using multiple context information.
Function***	(i) **Control** (of the system/application) or change in the environment by using an actuator. (ii) **Change status information** (or reconfiguration).
Discoverer	(i) Registration of the available components for the system as well as their capabilities and communication facilities (language, protocol, address, etc.). (ii) Determination of components that are no longer available in the system. (iii) Publishing information related to the components of the system by using (a) notification of changes or (b) polling.

*New component - **Modified functionality - ***Name changed

An Integrated Proposal.

Relations among Architecture Elements

- Complexity of C-A systems increasing due to interconnection of great amounts of heterogeneous devices and platforms

- C-A complexity does not introduce new relations among architecture elements in a substantial way

- Discoverer no longer a singleton in the architecture (API Gateway pattern)

  - **Hierarchy of discoverer components** to manage different and independent business subsystems due to technological, economical, performance, etc. reasons

Case Study

Health Risk Alarm

Case Study. Health Risk Alarm

- Context-aware system in the healthcare domain

- Detects emergency or illness situations affecting users by using contextual information gathered by sensors

- Alarm warnings and other similar actions based on user preferences, their contacts, or their geographical proximity among others

- Example: stress situation detection

Case Study. Health Risk Alarm

- Stress considered one of the autonomous mechanisms that allows human body to adapt to different demands

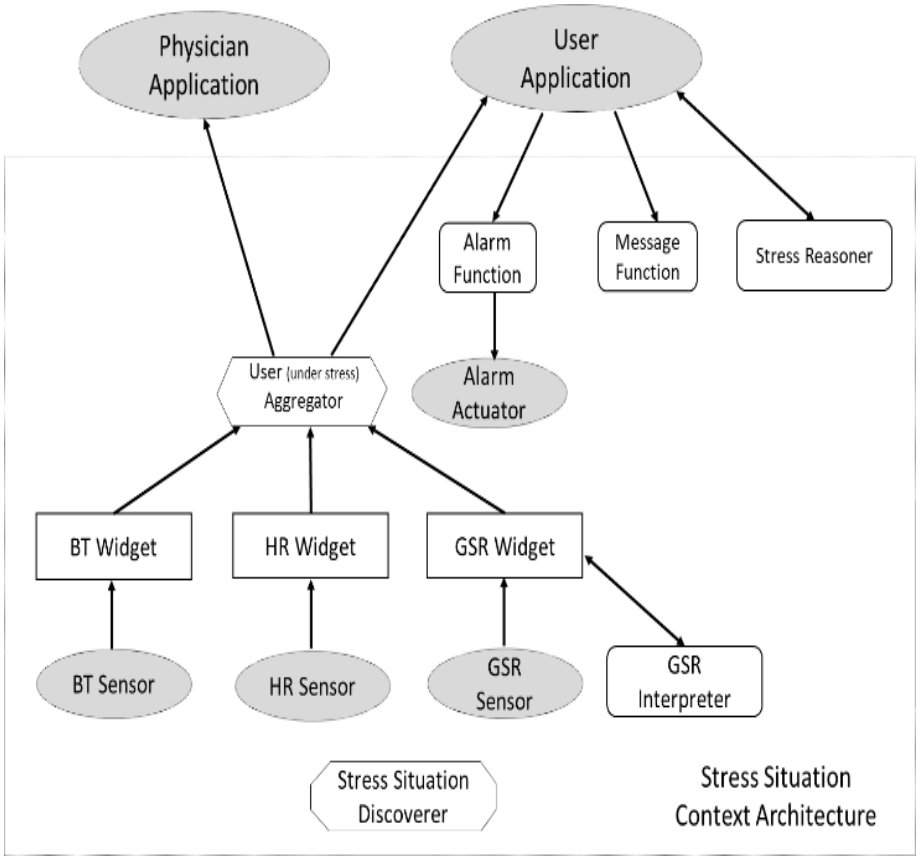
  - Frequent and prolonged exposure to high level stress can induce or exacerbate some cardiovascular or nervous deceases
  - Chronic stress drives to DNA damage advancing the ageing process, miscarriage, or cancer initiation

- Possible to measure stress episode intensity analyzing the variability of some physiological signals: Heart Rate (HR), Galvanic Skin Response (GSR) and, Body Temperature (BT)

Case Study. Health Risk Alarm

[EMERGENCY] “User has been under a high level of stress for a long time”

- Architecture configuration:
  - Includes all the abstractions defined
  - Supports the complete context life cycle mentioned (like the analyzed frameworks)
  - Satisfies C-A systems requirements defined
  - SoC degree increased (simple inference and reasoning supported by different abstractions)



Conclusions and Future Work

Conclusions and Future Work

- C-A systems are promising and challenging

- Architectural aspects play an important role to ensure and improve their overall quality

- New integrated proposal shown briefly in the design of a C-A system in healthcare domain

- More work needed for the development and evaluation of the presented framework to validate it in an empirical way

An Integrated Approach for Context-Aware Development

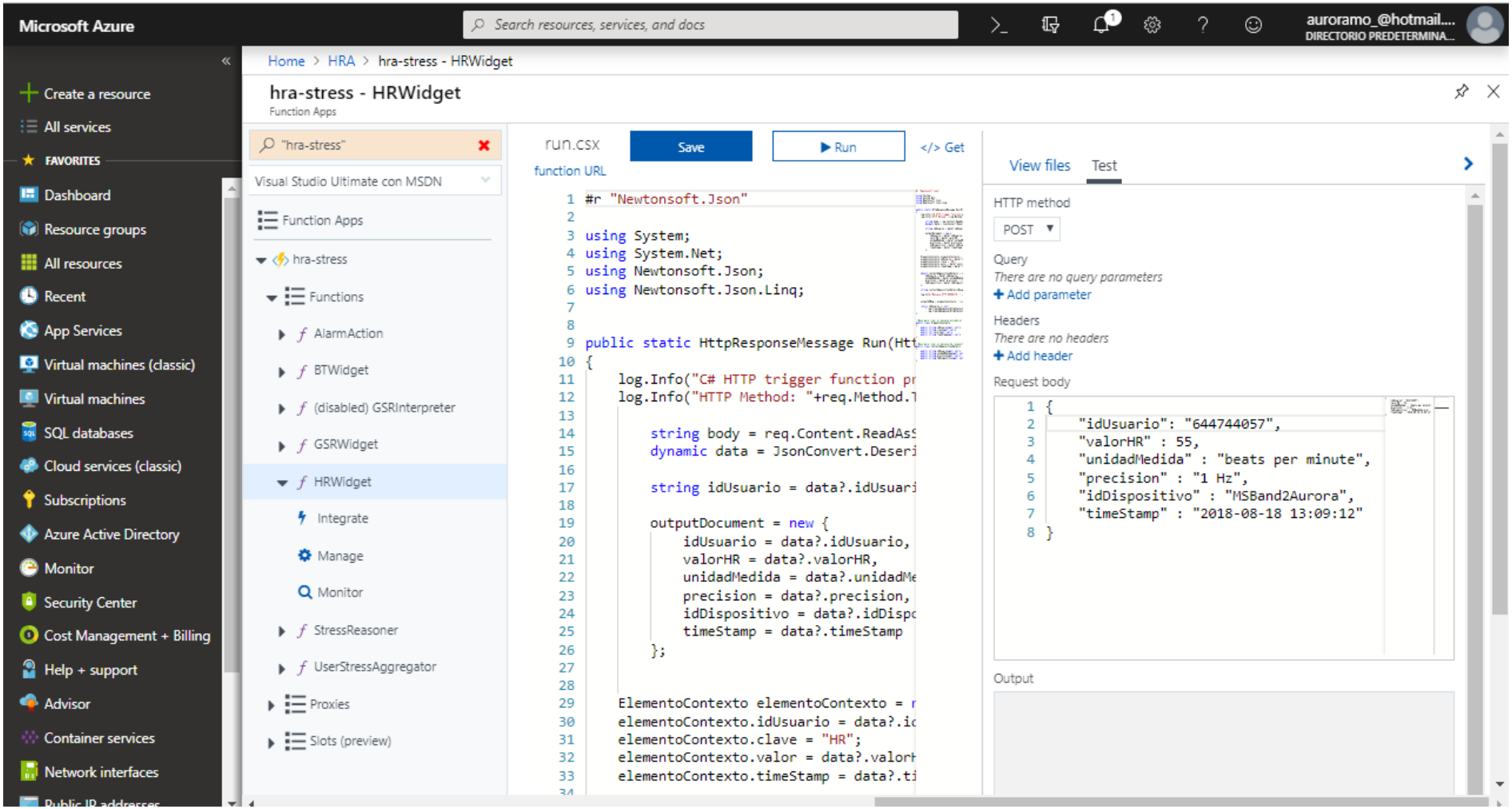
Aurora Macías – Indra Sistemas

Elena Navarro – University of Castilla-La Mancha

UCLM
UNIVERSIDAD DE CASTILLA-LA MANCHA

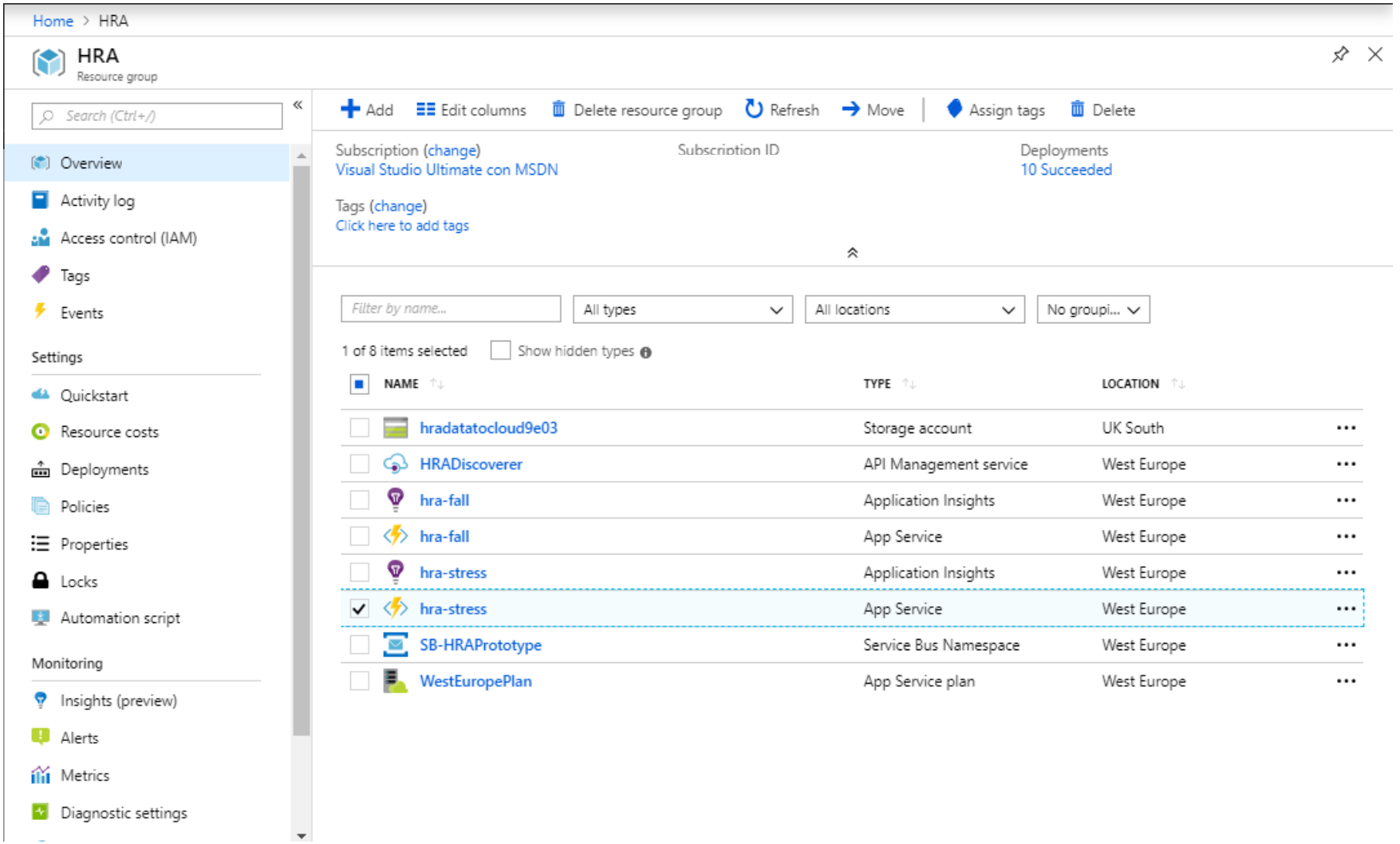
LoUISE
Laboratorio de Interacción con el Usuario e Ingeniería del Software

Case Study. Health Risk Alarm



Stress situation implementation based on the proposal and on microservices architecture

Case Study. Health Risk Alarm



Health Risk Alarm implementation based on the proposal and on microservices architecture