Master project
Winter term 2017-18

A Simulation Environment for a Dynamic SoS

Introduction

The ubiquity of smart devices, smartphones, RFID chips, and sensors, brings forth the formation of multiple diverse, interconnected networks virtually consisting of billions of components. Such a system comprises numerous smaller systems and is thus usually referred to a System of Systems (SoS). Both the design and operation of a SoS presents IT with exceptional challenges: how does an architect or a system engineer acquire even the basic understanding necessary to plan for possible interactions; the required fault-tolerance of critical components or even estimate the cost? Simulation can be a useful tool for addressing the extraordinary complexity of a SoS. Based on a model of the system at hand which accurately represents aspects of interest while abstracting from others; a simulation enacts specific realistic conditions and observes the model's reaction. The observations are then used to draw conclusions on how the real system would react under the same conditions. The simulation can be parameterized effectively leading to observations on many scenarios that would be difficult and costly to construct otherwise.

Simulation in Industry

Simulation is one of the key tools of various industries (e.g. automobile, manufacturing, healthcare and rescue) to understand the difficulty of the tasks at hand and facilitate the realization of difficult projects. The usefulness of simulation has led to the creation of very powerful simulation software (see production line example in Figure 1). However, the emergence of SoS introduces a new problem dimension. By their nature, some SoS are inherently dynamic; one should expect that the structure of these systems will change very frequently as numerous events occur. Furthermore, the components of such systems are not only based on software; components might be humans or physical objects that interact with the system. This behavior of the former cannot be fully predicted as is normally the case for software components.

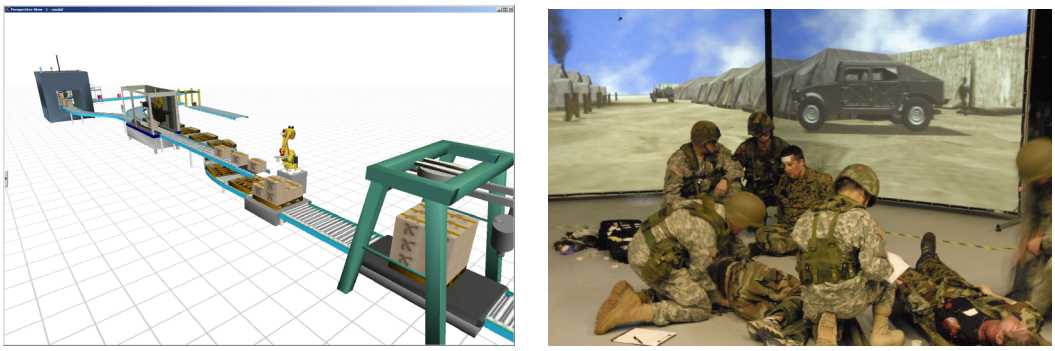

Figure 1 Simulation of a static environment (left[1]) and of a more dynamic situation (right[2]).

These characteristics lead to dynamic structures and runtime conditions that typical industry software has trouble simulating (for instance, the rescue in Figure 1). We plan to do research on describing these dynamic structures and conditions in a systematic way.

Application Domain

We base this master project on a use-case from the medical domain. Smart medical devices are considered a great prospect for the domain of healthcare services, specifically for continuous patient monitoring. The proliferation of these devices will result to a medical SoS where many physical components, smart devices, and clinicians will interact towards the goal of offering patients optimal services. Figure 2 shows an exemplary view of such an envisioned environment for a single patient; a smart device is attached to the patient. An example of a smart device in this context could be a device

[1] From http://www.flexsim.com

[2] From http://www.movesinstitute.org/healthcare

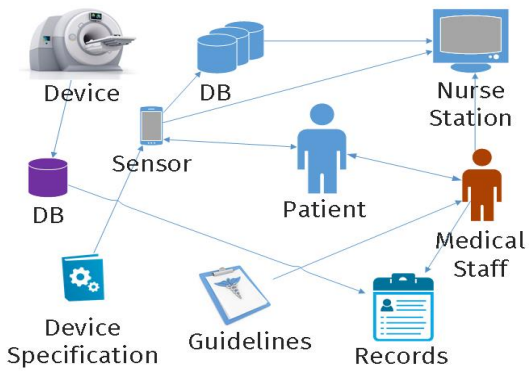

Figure 2 An Exemplary Medical Environment

that is supposed to take periodical measurements of a specific vital sign and, based on specific conditions, take an action, e.g. inform a clinician. The smart device stores its monitoring data to a database and sends it to the hospital's nurse station. Another device, e.g. EEG, that has been modified to be able to connect to this system, performs a test and store its data in another storage point. Meanwhile, a number of entities are expected, either directly or indirectly, to interact with the environment, e.g. clinicians who configure the devices, nurses at the nurse station who monitor the device's measurements, medical guidelines that provide instructions on required steps and procedures.

In this setting, a desired requirement would be to check, while the system is running, whether a medical process or a previous part of the system execution proceeded as intended. Another requirement is to timely detect situations that do not pose an immediate danger to the patient but could be dangerous conditionally, e.g. if a deadline for medicine administration is not met. We hold that providing clinicians with methods to satisfy these requirements will be crucial for the adoption of smart devices from healthcare.

In order to achieve this, we intend to exploit the data medical devices could generate (e.g. a periodical report on a patient's vital sign) and possible system events (e.g. device attached to or detached from a patient – connected or disconnected from the network). Our goal is to capture and aggregate this data and, by cross-checking it with medical guidelines, detect issues with the operation of the medical system or medical process. The cross-checking can either take place online, i.e. while the system is running, or offline, i.e. by capturing a trace of system events and replaying them at a later time.

Project Overview

As is the case in industry, in order to get a better glimpse at the complexity and difficulties of this task as well as testing our solutions, we need a simulation environment. The purpose of this master project is to create an environment that will simulate and visualize a fragment of a hospital system. This environment should consist of the following:

- A realistic model of a system, i.e. that will model a system consisting of multiple, heterogeneous devices
- The environment should be able to process a set of events, i.e. *an event trace*, e.g. incoming patient, device attached, device disconnected, etc., and adjust its structure and behavior accordingly.
- The environment should be able to *replay* a given trace, for instance, a trace describing a part of a standard medical procedure (taken from relevant guidelines).
- A graphical representation of the model and events, e.g. a "sensor disconnected" event will cause a component to disappear from the model's graphical representation.

The model's initial configuration parameters, e.g. type of device, will be able to be set using the environment's user interface.

Information

For the modeling tasks, we recommend the usage of standard technologies, i.e. JAVA Eclipse, IDE, and the Eclipse Modeling Framework (EMF). We expect interested students to have a basic level of familiarity with JAVA and Eclipse but we will provide an introduction to EMF (or further assistance on other technologies) when necessary. For questions and more information about this project, contact Holger Giese (holger.giese@hpi.de), Lucas Sakizloglou (lucas.sakizloglou@hpi.de).