

# A Novel User Interface for Controlling Industrial Environments and a Simulation Test Bed

Fraunhofer  
Institute for  
Intelligent  
Analysis and  
Information  
Systems (IAIS)  
Sankt Augustin,  
Germany

Maxim Foursa  
David D'Angelo  
Rejin Narayanan  
Georg Glock

*In this paper we present a novel user interface to industrial control systems and a simulation test bed. The interface and the test bed were developed within the framework of the INT-MANUS project. The main goal of the project is to develop a new technology for production plants, the Smart Connected Control Platform (SCCP). The platform is an open distributed learning agent-based system that integrates machines, robots and human personnel. The developed interface uses the SCCP functions and provides a new powerful tool, which can help operators of industrial plants to monitor and control their production processes.*

*Keywords: smart-connected-control platform (SCCP), simulation, mobile control system, graphical user interfaces*

## 1 Introduction

The European research project INT-MANUS addresses several problems of today's manufacturing [INT-MANUS; Foursa et al.

2006; Schlegel et al. 2007]. The project's aim is to develop the Smart Connected Control Platform (SCCP) for manufacturing enterprises. The platform allows controlling a production plant with the help of an open distributed learning agent platform, which integrates machines, robots and human personnel. The platform's architecture allows higher automation capabilities in production and forms a ubiquitous factory environment for human personnel. The interfaces to the platform should make it a "literally visible, effectively invisible" system [Weiser 1994], allowing operators to concentrate on most important tasks. The platform is described in [Schlegel et al. 2007] in detail.

In order to develop and test the above mentioned interfaces a test bed environment is needed. We have developed a miniature model, which simulates a processing line (from raw material delivery to quality control) and does not require much space.

In this paper we present the model factory (second chapter) and introduce the mobile control system (MCS) for industrial environments (third chapter).

## 2 FhG.IAIS Simulation Test Bed

In order to develop advanced industrial user interfaces in a laboratory environment, we designed and built a test bed that emulates industrial processes. The test bed is also used to demonstrate the concepts developed in the INT-MANUS project, such as decentralized communication, central supervisory control, interfaces for handheld and stationary computers and others. This chapter describes the design and implementation of the test bed.

### 2.1 Test Bed Hardware

Our goal was to develop a miniature model of a manufacturing plant, which includes a mobile robot and working machines. This would enable us to test our interfaces in the lab, without affecting a real factory. The layout of this test bed is shown in figure 1.

The test bed makes beams, using magnetic cubes. The four machines shown on figure 1 are: the Cube Dispenser (CD) (figure 2), Beam Maker (BM) (figure 3), Quality Controller (QC) (figure 4), the Delivery Bay (DB) and a miniature robot (R) (figure 5). Let us see a typical operation of the test bed. We start a program to manufacture a beam made up of 3 cubes. The Cube Dispenser ejects a cube and the robot carries it to the inlet of the Beam Maker, which takes in the cube, turns it on its side, and pushes it into the beam chamber. This task is repeated three times. Then the Beam Maker pushes out the finished beam. The robot carries it to the Quality Controller, where the beam is moved up a ramp and dropped. A beam that is not split up in this process is deemed good and is taken to the delivery bay by the robot. This completes the manufacture of the beam.

We chose to use the Khepera2 desktop robot [KHEPERA] based on our own survey of desktop robots. A gripper turret was mounted on the robot for transporting the cubes and beams. We decided to build our own model machines to suit our specific needs. The machines were fabricated at our institute where a well equipped workshop is available.

The machines were fabricated using aluminium X-form beams, which add flexibility to their mechanical structure. Machine parts fabricated in our workshop were made out of aluminium and polyoxymethylene. SolidWorks [Solidworks] and

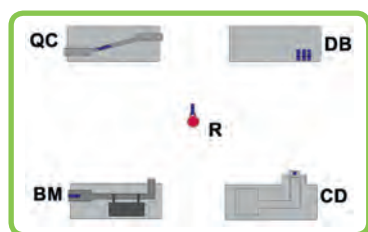


Figure 1:  
The Layout of the  
Miniature Test Bed

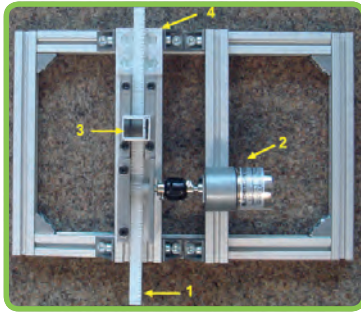


Figure 2:  
CD – Cube Dispenser Machine  
1 – Piston Assembly  
2 – Motor  
3 – Magazine for Cubes  
4 – Cube Delivery Platform

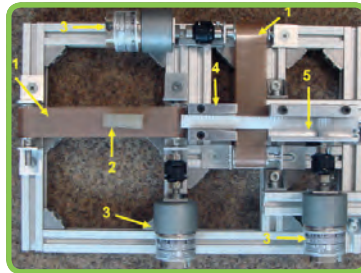


Figure 3:  
BM – Beam Maker Machine  
1 – Conveyor Belts with Light Barriers  
2 – A Beam  
3 – Motors  
4 – Cube Chamber  
5 – Piston Assembly

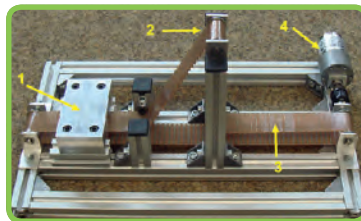


Figure 4:  
QC – Quality Controller Machine  
1 – Beam Aligning Device  
2 – Ramp  
3 – Moving Belt with Light Barrier  
4 – Motor

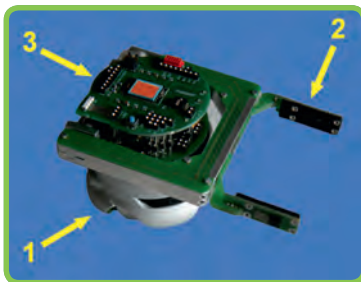


Figure 5:  
Khepera Robot  
1 – Chassis  
2 – Gripper Turret  
3 – Communication Turret

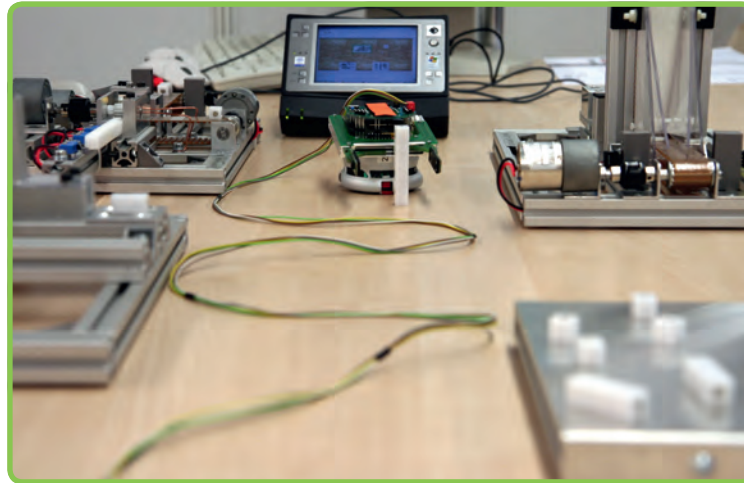


Figure 6:  
The Test Bed

hand sketches were used for designing the machines.

Building the magnetic cubes posed many challenges. The robot's gripper can lift only 50g and our beam are made of 3 cubes. The beam would slip from the gripper if the surface were polished. The chamber of the Beam Maker machine needs the surface of the cubes to be smooth so that they slide inside it. Our fabrication tools would work only with metals. The optimal choice for these requirements was found to be cubes of 10mm sides made from polyoxymethylene. Similarly, the magnets used had to be strong enough to keep the beam together when it falls from the Quality Controller, and not too strong that the cube would stick to the metal rods inside the robot's gripper turret. We found 2x10x6mm nickel plated neodymium magnets to be the best suited ones for the cubes.

The machines have sensors like magnetic switches, light barriers and mechanical position switches. They also have actuators like pistons and conveyor belts, which are driven by 12V DC electric motors with a gear ratio of 100:1. These motors and sensors are connected to the electronics on each machine, and these are in turn connected to a central embedded system for the entire test bed. This system, a single board GNU/Linux computer DiL/NetPC [DiL/NETPC],

runs atomic actions on the machines, and communicates with the test bed control software via a TCP/IP interface. figure 6 shows the test bed.

The detailed description of the machines is presented in figures 1 to 6.

Besides the model factory, we have set up other systems in our laboratory: Ubisense position tracking system [Ubisense] for mobile devices and an optical tracking system for estimation of the robot's position. The former is a commercial system, which uses ultra-wideband (UWB) technology and locates objects (so called "tags") in 3D with an accuracy of 15cm. The latter is our own one camera-based 2D tracking system developed on the basis of OpenCV [OpenCV], which is able to track the position of the robot on the test bed table with an accuracy of 0.5 mm.

## 2.2 Control Software

The structure of the control software of the test bed is shown in figure 7.

The test bed described in the previous chapter is shown on the Hardware layer. It is controlled by the Test bed Control System (TCS) software written in C++, which is presented in the Production processes control layer. The software communicates with the test bed

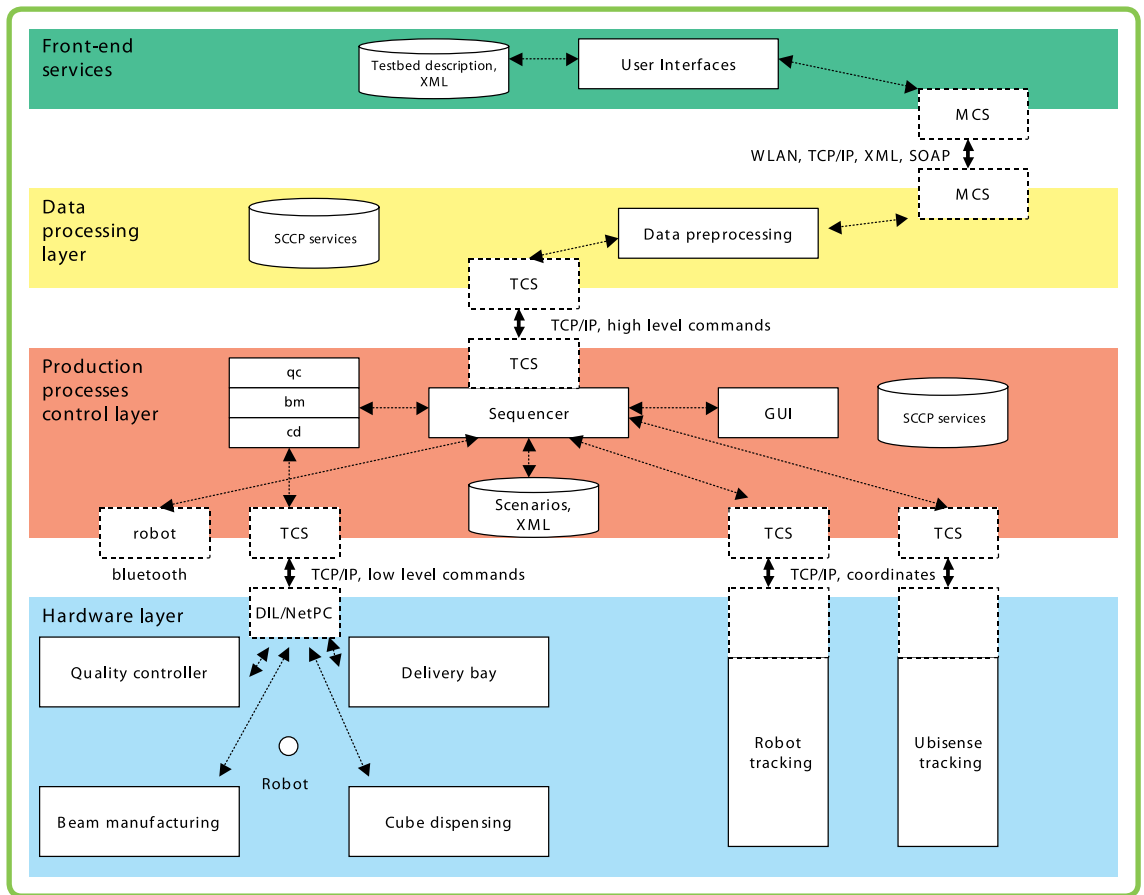


Figure 7: The Structure of the Control Software

embedded system via TCP/IP, with the robot via a serial link or Bluetooth [Bluetooth] and with other objects via SOAP protocol [GSOAP]. The TCS can read test bed programs from xml files, which describe entire production scenarios, and execute instructions for the machines as well as the robot. Thus, the XML-based program file forms a script-like set of instructions for the entire test bed. The TCS also receives updates from the tracking systems and uses them in the scenarios or forwards them to the Mobile Control System.

The Mobile Control System (MCS) on the front-end layer is configured for the handheld computer SONY VAIO U8 [Sony Vaio PC] (see figure 6). It provides a graphical interface for a user to collect information about the model factory and to interact with it. The MCS is described in detail in the next chapter.

The communication between the TCS and the MCS is done via a network abstraction layer. This separation allows using a universal data format. This approach simplifies integration problems with a new industrial environment, because only changes to the data processing modules are required, while the core UI modules remain the same.

The TCS is responsible for registration and management of Mobile Control Systems. After an MCS is successfully registered, it will be informed by the TCS every time a change of the environment (an event) occurs. Another task of the TCS is the distribution of updates and semantic messages to all registered MCS. The MCS is able to send arbitrary commands to the TCS. These commands will be processed by the TCS and passed to the environment specific production control system.

The TCS provides two different networking interfaces. The first interface is responsible for the connection to the system that controls the production machines. This system is not necessarily running on the same machine; thus this interface encapsulates the core TCS functionality from the environment specific machine commands. The second interface defines the network connection between the Test bed Control System and the Mobile Control System. This communication must satisfy different needs for different production environments (i.e. encryption and real-time functionality).

While the Hardware layer represents the peer structure of the SCCP described in the second chapter, the Production processes control layer represents the SCCP higher-level services for the test bed. The test bed can be considered as a cell in the industrial hierarchical description,

with its specific resources and tasks.

### 3 Mobile Control System

As a primary mobile device we have chosen the SONY VAIO U8 handheld computer [Sony Vaio PC] (see figure 6). This computer is slightly bigger (167x108x26.4 mm) and heavier (550 g) than an ordinary PDA, but it features a 800x600 SVGA touch panel, 900 MHz Celeron processor, USB 2.0 interface and full Windows compatibility. The computer also has wireless LAN support (IEEE 802.11b/802.11g).

The requirements to our Mobile Control System are the following. The system should provide a simple and intuitive access to the operator's workplace. Even without any knowledge of our system an operator should be able to connect to different machines and interact with them. At the same time the system should allow visualization of the whole industrial environment the operator has to control. It should also be able to visualize different notifications and messages coming from SCCP services.

For all manufacturing machines and other objects (for example, the localization system) we have a list of sensors, actuators and commands. Sensors passively report their statuses as values, actuators - as states. SCCP analyses the statuses and states according to the context and adds information about their meaning, for example, as notations "good", "average" or "dangerous". The commands are supported by the SCCP services or machines themselves and can be executed by the operator.

The whole industrial environment that has to be visualized is described in an xml file. The description includes positions of the machines in the room and the above mentioned lists of sensors, actuators and com-

mands for each machine or object. The xml file has to be loaded by the MCS software before its execution.

We visualize the machines as rectangles and robots or mobile devices as circles. According to the SCCP's semantic information ("good", "average" or "dangerous"), the status of the sensors is visualized as green, yellow or red square on the "global view" screen. Additional information, as the name of current part program, the degree of fabrication or current status of the machine can be also visualised. Based on this information, SCCP also derives the status of machines and other objects. The status of machines can have the following states: "working", "failure", "average/attention required" and "switched off/scheduled for maintenance". The status is visualised as green, red, yellow or black border around the machine's rectangle. And, finally, the SCCP analyses the status of the cell and the MCS visualizes it as a circle in the top-left corner. For robots actual routes can be shown; for tracked robots and mobile devices - actual positions.

Besides the interfaces, the MCS is also responsible for user notification deliveries. In case of an error, the SCCP sends notifications to all registered MCS and operators can react more efficiently. SCCP can also track maintenance schedule and notify operators when maintenance has to be done.

In our approach we combine three different views (operation modes) in one interface. The information displayed in a view is a mixture of generic data available for all objects inside the SCCP and special data only available for single objects or object groups. The operator can switch easily between the modes and turn on/off full-screen mode. The three main operation modes are as follows:

- › 1. Global View: This view shows an interactive map, where the position of users, robots, and machines can be seen (see figure 8). To simplify the recognition and provide basic information about the status of the production environment, all objects are displayed as pictures with colour coded information showing their statuses. The operator can zoom in/out, execute specific commands for the whole cell and move the map by pointing and moving the stylus to the desired direction. By double-clicking a machine the operator enters the local view mode.
- › 2. Local View: This view offers a user the possibility to get more detailed information about a chosen object, like sensor and actuator data. Using this view, the user can also send object specific commands to the hardware layer over SCCP services. For a real manufacturing machine it can be designed similarly to the numerical control screen (see figure 9). Coming back to our testbed, we consider the light barriers of machines as sensors, motor controllers as actuators. Each machine is able to send its parameters to any network peer, which may be another machine or the MCS. Moreover, each machine has a list of supported commands. For example, for the BM machine the commands are: MoveBeamOut, MoveCubeIn, PushBeamToBelt, PushCubeToChamber (see figure 10).
- › 3. Product View: In this mode the operator is able to see a geometrical model of a part/product and send a command to the SCCP for its production. With the use of an external miniature camera, which can be connected via USB to the handheld computer, all produced ob-

jects can be photographed. The operator can visualize a geometrical model of the product and overlay it with the live image. He can then compare the produced object with its model thus controlling the production processes on-site not using any special measurement tools.

The MCS software is implemented in C++ with Qt [Qt toolkit] toolkit for MS Windows platform. The software is fully

integrated with the test bed described in the previous chapter. Adaptation of the software for another industrial environment or Windows-compatible computer can be done easily.

### 5 Conclusion

In this paper we presented a new approach to industrial control systems. We described the Smart Connected Control Platform for manufacturing enterprises, which allows controlling

a production plant with the help of an open distributed learning agent platform and forms ubiquitous computing environment in factories. The services provided by the platform allow industrial operators and managers to monitor their environment and supervise production processes with high efficiency. We described our simulation test bed and presented the mobile control system for handheld computers as an interface to the SCCP services for shop-floor operators.

Acknowledgments. The INT-MANUS project is sponsored by the European Commission under the contract number NMP2-CT-2005-016550.

### 6 References

Bluetooth:  
<http://www.bluetooth.org/>

DiL/NETPC family:  
<http://www.dilnetpc.com/>

Foursa, M.; Schlegel, T.; Meo, F.; Herrmann Praturlon, A.; Ibarbia, J.; Kopacsi, S.; Mezgar, I.; Salle, D.; Hasenbrink, F.: *INT-MANUS: Revolutionary control of production processes*. ISBN:1-59593-364-6, ACM SIGGRAPH 2006

GSOAP:  
<http://www.cs.fsu.edu/~engelen/soap.html>

INT-MANUS project web-site:  
<http://www.int-manus.org>

KHEPERA robot:  
<http://www.k-team.com/>

Meo, F.; Foursa, M.; Kopacsi, S.; Schlegel, T.: *Use of agents for predictive maintenance and diagnostics of machine tools*, CIRP ICME 2006, 25–28 July, Ischia, Italy, 2006

Morley, Richard E.; Parunak, H.; Van Dyke: *Autonomous Agents and Chaos in Daily Life and Manufacturing Systems*, International Control Engineering Conference, March 14–16, 1994

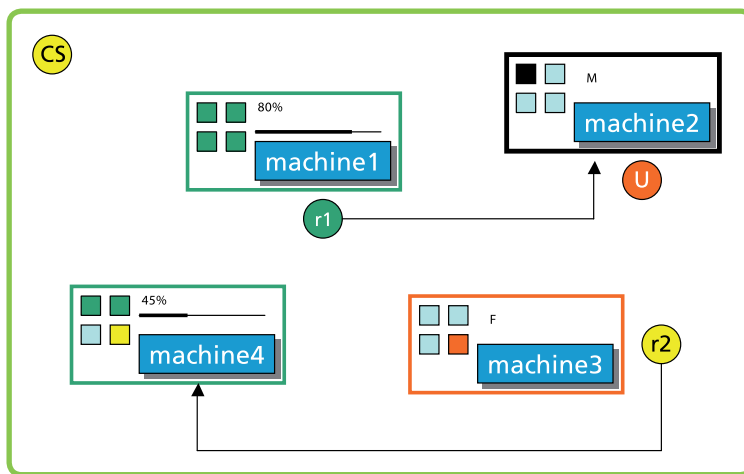


Figure 8: Schematic Representation of a Cell in the "Global View" Mode

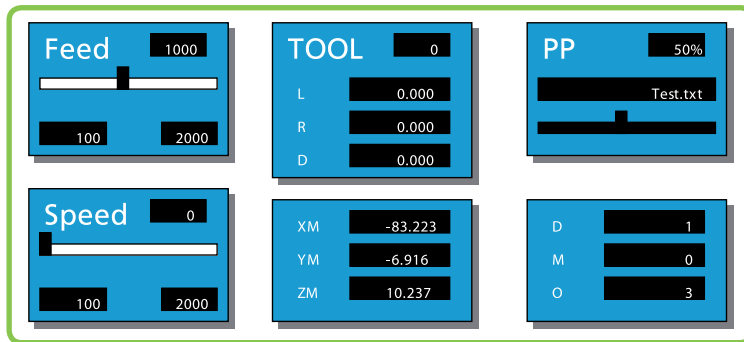


Figure 9: Schematic Representation of a Machine in the "Local View" Mode

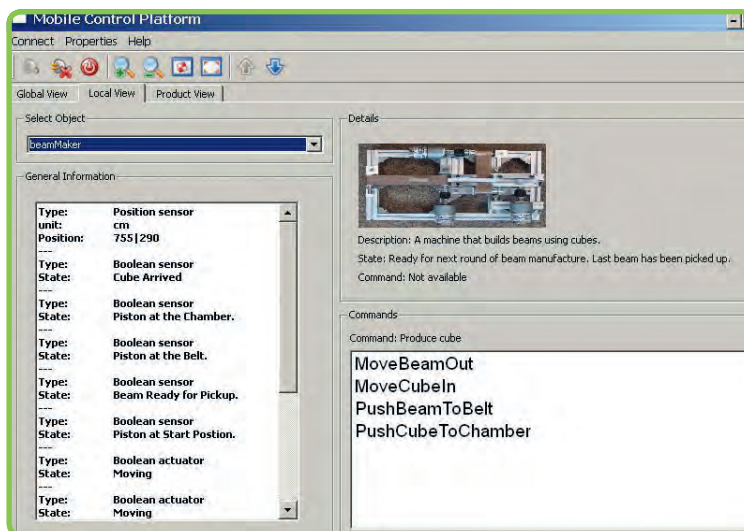


Figure 10: "Local View" Mode of a Test Bed Machine

OpenCV:  
<http://www.intel.com/technology/computing/opencv/>

Pinto, J.: *The Truly Distributed Control Revolution*. ISA Proceedings, 94 and Intech July, 95, Web-version updated Dec. 99; <http://www.jimpinto.com/writings/ioio.html>

Qt toolkit:  
<http://www.trolltech.com/products/qt/>

Schlegel, T.; Srinivasan, A.; Foursa, M.; Bogen, M.; Haidegger, G.; Mezgar, I.; Canou, J.; Salle, D.; Meo, F.; Agirre Ibarbia, J.; Herrmann Praturlon, A.: *INT-MANUS: Interactive Production Control in a Distributed Environment*, accepted to International Conference on Human-Computer Interaction (HCI) 2007 conference, Beijing, China, 22–28 July 2007

Solidworks:  
<http://www.solidworks.com/>

Sony Vaio PC:  
[http://vaio-online.sony.com/prod\\_info/vgn-u8g/specifications.html](http://vaio-online.sony.com/prod_info/vgn-u8g/specifications.html)

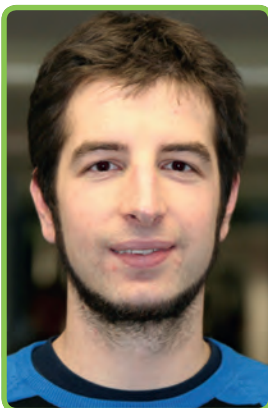
Ubisense:  
<http://www.ubisense.net/>

Weiser, M.: *Building Invisible Interfaces*. Computer Science Lab. Xerox PARC. UIST94, November 1, 1994



Maxim V. Foursa  
[maxim.foursa@iais.fraunhofer.de](mailto:maxim.foursa@iais.fraunhofer.de)

Maxim Foursa is a project manager and research scientist at the CC VE IAIS. He has a professional background in management and information science with working experience of over seven years. He currently coordinates the EU project INT-MANUS.



David D'Angelo  
[david.d-angelo@iais.fraunhofer.de](mailto:david.d-angelo@iais.fraunhofer.de)

In 2005 David D'Angelo received a bachelor degree in Computer Science from the University of Applied Science Bonn-Rhein-Sieg. Currently he is working on his Master thesis about Interaction Techniques in Virtual Environments.



Rejin Narayanan  
[rejin@arssoftware.com](mailto:rejin@arssoftware.com)

Rejin Narayanan received the B.Tech degree in Electrical and Electronics Engineering from the University of Kerala, Thiruvananthapuram, India, in 2000, and the M.Sc. degree in Autonomous Systems from the Bonn-Rhein-Sieg University of Applied Sciences in 2007. From 2006 to 2007 he worked on the INT-MANUS project at the Fraunhofer IAIS.



Georg Glock  
[glock@trackmen.de](mailto:glock@trackmen.de)

Georg Glock worked as a technical employee at GMD and Fraunhofer IAIS for more than 10 years. He worked in the INT-MANUS project from January 2006 to October 2007 as a technical coordinator.