# REXOS 3.0 PERFORMANCE AND RECONFIGURABILITY PROJECT PLAN

Daniel Telgen
University of Applied Sciences Utrecht

Alexander Streng - 1568055
Duncan Jenkins - 1574726
Roy Scheefhals - 1563303

October 4, 2013

# Contents

## Versioning

| Version | When | Who | What |
|---|---|---|---|
| 0.1 | 23-09-2013 | Duncan Jenkins | Created initial document, created headings. |
| 0.2 | 28-09-2013 | Duncan Jenkins, Roy Scheefhals | Made project description |
| 0.3 | 01-10-2013 | Alexander Streng | Overhauled document to LaTeX and redone sections |
| 0.4 | 02-10-2013 | Alexander Streng, Duncan Jenkins, Roy Scheefhals | First version of document with all sections |
| 0.5 | 04-10-2013 | Alexander Streng, Duncan Jenkins, Roy Scheefhals | After proofreading, fixed small errors and oversights. Added section methods and tools. |
| 0.6 | 04-10-2013 | Alexander Streng, Duncan Jenkins, Roy Scheefhals | Processed project supervisor feedback into the document. |

# i Acronyms

**ROS** Robot operating system

**MAS** Multi-agent system

**MAST** MAchine STate

**GM** Grid manufacturing

**RMS** Reconfigurable Manufacturing Systems

**HU** Hogeschool Utrecht, this is the dutch name for the University of Applied Sciences Utrecht

**JADE** Java Agent DEvelopment framework, for more information see section 5

# ii Definition of terms

## Agents

One of the most commonly known definitions of an agent is as defined by Wooldridge [1] *"An agent is an encapsulated computer system that is situated in some environment and that is capable of flexible, autonomous action in that environment in order to meet its design objectives."*

## REXOS

REXOS is abbreviated from *REQS* to be short for Reconfigurable equiplets.

## MAS

A proper definition of an multi-agent system is as said by Zhang; *"A multiagent system is a distributed system consisting of multiple software agents, which form a loosely coupled network, called a multiagent system (MAS), to work together to solve problems that are beyond their individual capabilities or knowledge of each entity."*[2].

## Grid

An collection of equiplets that are able to produce in parallel is called an grid.[10]

## Reconfigurable Manufacturing Systems

According to Y. Koren RMS consists of several reconfigurable components; *"A Reconfigurable Manufacturing System (RMS) is designed at the outset for rapid change in structure, as well as in hardware and software components, in order to quickly adjust production capacity and functionality within a part family in response to sudden changes in market or in regulatory requirements."* [3]

## MAST

MAST stands for 'Machine States' of equiplets in REXOS. In order to use the equiplets properly and safe, proper machine states have to be implemented. A previous project researched this topic and had results in a paper[5].

## ROS

Ros is an operating system targeted at providing core abilities to develop code for robots. According to their own site *"ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license."*[1]

## Equiplet

In project REXOS all RMS are modular and can be reconfigured. They are named *equiplets* [4]. Equipltes are special production platforms that are cheap, agile and easy configurable. These platforms can operate in parallel.

## ROS

Ros is an operating system targeted at providing core abilities to develop code for robots. *"ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more. ROS is licensed under an open source, BSD license."*[11]

## Product Agent

Every product that is being is virtually represented in software by an autonomous entity. In the MAS system it is implemented as a 'Product Agent' [5]. The product agent will monitor and guide the lifecycle of a product.

## Module

*"each of a set of standardized parts or independent units that can be used to construct a more complex structure"*[2] An module within ROS is a specific piece of hardware equipment used to perform specific actions. (eg. a pen module is used to print dots)

## Node

Nodes are processes that perform computation. ROS is designed to be modular at a fine-grained scale: a system is typically comprised of many nodes. In this context, the term "node" is interchangable with "software module."[11]

## Stewart platform

The Stewart Platform mechanism, originally suggested by D. Stewart in [1], is a parallel kinematic structure that can be used as a basis for controlled motion with 6 degrees of freedom (dof).

[1] http://wiki.ros.org/
[2] oxford dictionary

### Product / service / equiplet step

A representation of an product within the REXOS platform consists of 'product' steps. These steps represent the various steps needed to completely construct a product. These steps are used to create service steps, a less abstract step used to calculate production time needed. Lastly these service steps are used to create equiplet steps, which are instructions for the equiplet.

# Introduction

## 1 Background

In 2008 a project called HUniversal Production was started by E. Puik. This project was started to research agile manufacturing[6] and to develop a working platform based on this manufacturing paradigm. Since then, the research into agile manufacturing became larger and a lot more research was needed on different parts of the whole platform.

The first prototype of the platform consisted of a single machine capable of picking up and placing different kind of objects. While developing this first prototype a lot of focus was in the architecture of the platform. The architecture was based on a hybrid architecture consisting of ROS and JADE.

In 2012, the idea of reconfigurable equiplets was introduced. Being able to safely remove, replace or add new modules to a running system. This led to the development of MAST, or MAchine STates. Using MAST simplifies checking whether a module is busy or idle, but more importantly, it allows modules to be put in a safe state, making removal possible.

Currently, the platform uses MAS and ROS, which communicate through blackboards to create an agile manufacturing system. Reconfigurability has not currently been implemented. Also, the performance of the current platform is unknown. This leads to the assignment at hand. More about the actual assignment will be explained later in this document.

## 2 The organization

The organization through which this assignment was offered is the University of Applied Sciences Utrecht (HU). More specifically, the department for Microsystemtechnology and Embedded Systems (MST). This department involves itself with product-improvement and the industrialization of microsystems and can be found in the Knowledge Center for Technology and Innovation at Oudenoord 700 in Utrecht.

In this department, teachers, also known as lectors, work together with research teams to conduct research at a practical level. Their goal is to provide professionals with solutions when they run into problems in the field. The area of expertise for the MST department focuses on high-precision systems.

Research within the HU in general targets innovation and is closely connected to the education provided. Important aspects are knowledge and the distribution of knowledge. This is why the HU conducts research on a practical level.

# The project

## 1 Project description

The machines used for production in factories today range in the hundreds of thousands of euros. These machines are highly specialized for specific tasks and are usually not adaptable to their environment or needs. This creates a very inefficient setup for the manufacturers. Because of this, the initial project was created to come up with a cheap and reusable alternative to modern day factory machines. The resulting idea from that project was to create a grid of machines, each capable of a task depending on which modules are attached to the machine. These machines are called equiplets. This way of having a manufacturing floor is called Agile manufacturing'[8].

One of the major drawbacks of current machines is that they are suited to only a single specific task. Therefore one of the aspects of this project includes reconfigurability for the equiplets. The idea is to be able to attach and detach modules from the system without first needing to shut it down. This means the system will detect changes in its modules and adapt to those changes. This in turn means that the production machine can be reconfigured on-the-fly to allow for the creation of new products without a long period of downtime.

The focus of agile manufacturing is to create a manufacturing grid that can produce a large variety of products with a relatively low amount. Also the goal is to make relatively cheap manufacturing robots ( equiplets ) and modules.

To successfully create such a platform there has to be a solid architecture. As stated in the paper about this question[9], a hybrid architecture consisting of ROS and JADE can be the solution. This new platform in development is called REXOS. The research that will be done as described in this document will be beneficial to the REXOS platform.

## 2 Project organization

This research project consists of 3 team members and one project supervisor. The research project has close relations with other projects beneficial to REXOS. There is a team of interns that is developing the vision part (vision team). There is another research team researching the logistics part of REXOS. The research team conducting the research described in this document has close collaboration with the vision team, because the two teams are using the same hardware ( the equiplet ). The people involved in the research project are:

The researchers involved in the project are students at the university in Utrecht. They work directly under D. Telgen in a researching capacity.

| Name | Role |
|---|---|
| E. Puik | Founding father |
| D. Telgen | Project supervisor |
| A. Streng | Team supervisor, researcher/software developer and scrum master |
| D. Jenkins | researcher/software developer |
| R. Scheefhals | researcher/software developer |
| A. Hustinx | Team supervisor, intern/software developer and scrum master |
| T. Bakker | intern/software developer |
| G. Hakopian | intern/software developer |

Table 2.1: Project team

## 2.1 Scrum

As a projectmanagement method *Scrum*[13] will be used. Scrum is an *Agile projectmanagement method*. In Scrum, 'Agile' means that the enduser or project supervizor can change certain aspect of the project while the project is in progress. The Scrum method supports this kind of workflow.

The work that has to be done are called stories in Scrum. These stories are short worktasks that mostly can be finished within one day. These stories are stored in a so called *Product Backlog*. These stories will be given an estimation of how long it will take to finish the story.

In Scrum, project time is being divided in iterations of fixed lengths. These iterations are called *sprints*. In this project sprints will be planned of a length of one week. When a sprint is being planned, stories from the Product Backlog will be picked. which stories are being picked depends on what the goal is of the sprint and how much work can be done by the team. Stories that could not be completed in that sprint will be put back into the product backlog so they can be planned into a new sprint.

In this project, the digital Scrumboard *Assembla*[1] will be used. Assembla is a Scrum tool that provides a good overview of all Scrum functionality.

Considering this project is a research project that can change a lot, Scrum will be the best method.

## 2.2 Communication

In this project Daniel Telgen is the project supervisor. While conducting the research, all communication about the research being done goes through the project supervisor and there will be weekly meetings. In these meetings the progress will be discussed and new assignments will be planned for the next week in the form of sprints. The project leader wil also be available daily through mail.

A time sheet will be made during the project to make sure that each team member is present. This has to be done since some members need a day off every week for other obligations.

# 3 Current state of the project

Up until this moment the development of project REXOS was focused on the initial architecture and functionality of the equiplets. In this stage of the project the focus is drawn towards what the current performance of REXOS is and how reconfigurability can be implemented in REXOS.

[1] `https://www.assembla.com/`

At this moment only one equiplet run the required software and contains the hardware to test performance. This means that data regarding the performance of multiple equiplets cannot be gathered, however, more equiplets are expected to arrive soon. Until these extra equiplets arrive, meaningful test data cannot be gathered. Initially there was some difficulty getting the platform to run in a stable manner. The problems causing this have been solved however, which now allows for proper testing to be conducted.

A system for gathering testing data has been partially implemented into the platform. This system gathers information about the communication between agents on the platform. Communication between other parts of the platform is not implemented yet however. This implementation will need to be completed before performance testing starts.

Reconfigurability has not been implemented at all, although the system does use MAST.

# 4 Project activities

During this project we will mainly focus on three major activities. Gaining a **stable platform**, gathering a wide variaty of **performance metrics** and implementing / researching **reconfigurability**. The stable platform is needed to continue working on this project. Not only for the current group of researchers / interns working on the project, but for the next generation of students who will participate in this project aswell. The performance metrics are of great importance because they are needed for an extension of the FAIM[5] paper in a journal. Reconfigurability is the main focus of our research project. By researching (and possibly implementing) reconfigurability, the project would get a major boost. It is also a core feature of the current system to have reconfigurability at its disposal.

## 4.1 Phases

Due to a upcoming deadline for the precision convention[2] there are two different phases that are slightly parallel to each other. The main phase, in which reconfigurability is planned, is defined on the wiki[3]. The other phase, which is defined because of the precision convention, is mainly focusing on gaining a stable platform and writing some demonstrators. After both those phases there is one more phase in which we will finish implementing reconfiguration.

- Performance phase
  This main phase will focus on writing code to get the performance metrics. Keeping the second goal of creating a stable system in mind, some work on the stability of the platform will be done. In order to get these performance metrics, specific testing code has to be written and researched. A means to gather this data (e.c. logging) has to be implemented as well.

- Precision phase
  The phase leading up to the precision convention will mainly be about stabilizing the platform and writing demonstrator code. This phase is almost completely parallel to the Reconfigurability phase. Due to the upcoming precision convention, a total of 3 demonstrators have to be written. The whole platform itself, including making of a product, planning and producing of the product has to work as well.

[2] http://www.precisiebeurs.nl/
[3] http://wiki.agilemanufacturing.nl/index.php/REX_3.0_Phase_1

- Reconfigurability phase
  This phase will be all reconfiguring. This phase starts after the precision fair and will last untill the end of the research semester. Due to the nature of the project reconfiguring is an important aspect. During the first two phases the interns will have done most work on implementing reconfigurability on a lower level thus allowing research to be done on the intelligence behind reconfigurability. This means researching the best way to deal with changing modules and hardware on the MAS level.

## 4.2 Milestones

Throughout the project several important milestones are defined. It is clear that all of these need to be achieved in order for our research to be succesfull (maybe with exception of the demonstrators, but these are quite important to the HU).

- **Stable platform**
  In order for this project to work, the platform on which it is designed needs to be stable. This is currently not the case. Most of the system crashes whenever unsuspected conditions are present and there are still features that need to be implemented. The most important features are:

  - *Handle multiple equiplets*
    Currently the platform isn't capable of handling more then one equiplet. As soon as multiple equiplets are present within the system, it overloads[4]. Part of the solution is to test and search for the cause of this problem and solve it.

  - *Handle multiple product agents*
    Whenever multiple product agents are present within the grid, the planning of product steps tends to mismatch. Research is needed as to why this is happening, and a proper solution has to be implemented.

- **Performance metrics**
  Due to an upcoming deadline for a journal, there is a need to know certain metrics of the (stable) system. These need to be gathered and parsed into something usable. Determining which metrics should be gathered has been done and only implementing the code to do so remains. These metrics are:

  - *The amount of JADE messages sent*
  - *JADE messages that have been lost / not received*
  - *number of timeouts between JADE message communications*
  - *ROS messages are being sent through an actionserver*
  - *The IO to and from the MongoDB blackboards. Fetch speed/ process time / tailable cursor speeds*

  In order to gather sensible data, test scenario's will have to be constructed. Code needs to be implemented as well to gather the data, and process it in usefull statistics.

[4]concluded after several tests

- **Reconfigurability**
  Reconfigurability is an important aspect of the platform[3] but not yet implemented. Once reconfigurability on a modular level has been achieved[5], the reconfigurability on the MAS level has be researched and implemented. This means doing research on how to make the software agents 'aware' of their modules, and how to let them act when one changes. The intelligence should be able to deal with unsuspected conditions, malfunctions and human errors.

- **Demo's**
  Due to the upcoming precision convention, demonstrators need to work and the platform has to be stable. During this convention, the goal is to show a demo of the completely functional system. The setup on the convention will be as following:

  - *Two pick and place robots*
  - *3d printer*
  - *Stewart platform delta robot*

  All these equiplets will be placed within the same grid, and will operate on the REXOS platform. This means that the platform should be fully functional at this time. Demonstrator code needs to be written as well to guarantee smooth demo's.

### 4.3 Planning

Due to the abstract nature and timing of the phases, the planning will have some parallel elements. Details of the planning will be done "on demand" depending on Customer demand. According to the SCRUM method.

| **Phase** | **Date span** | **Description** |
|---|---|---|
| performance phase | 21-08-2013 - 03-12-2013 | Stabilize platform, gain performance metrics |
| precision phase | 29-09-2013 - 03-12-2013 | performance metrics and stabilize platform. Write demonstrators. |
| reconfigurability phase | 03-12-2013 - 01-02-2014 | research and implement reconfigurability |

Table 2.2: Project phases

## 5 Methods and tools

### 5.1 Jade

Java Agent DEvelopment Framework is a software framework to develop distributed agent-based applications in compliance with the FIPA[6] specifications for interoperable intelligent multi-agent systems. An application based on JADE is composed of a set of components called "agents" implementing the pieces of functionality required by the application. JADE primarily provides the Agent and Behaviour (a task to be executed by an agent) abstractions, transparent distribution of agents accross a wide range of devices, peer-to-peer communication between agents and

[5]This is done by the interns
[6]http://www.fipa.org

a publish-subscribe discovery mechanisms that allows agents finding each other. Furthermore JADE provides a number of additional features such as agent mobility, ontologies and content language support, fault tolerance and web services integration and a rich suite of graphical tools that facilitate the administration of a JADE based application. JADE provides:

- An environment where JADE agents are executed.
- Class Libraries to create agents using heritage and redefinition of behaviors.
- A graphical toolkit to monitoring and managing the platform of Intelligent Agent agents.

### 5.2 ROS

ROS is the main platform used to create and run code for modules (called nodes in ROS). ROS is an multi-langual, peer-to-peer, tool-based, thin and open source framework for robotics software[11]. ROS can be used as an abstraction layer for the hardware in modular machine and robots. ROS utilizes the C++ program language (as well as 3 others[11]) and can be used directly to control hardware systems in realtime.

### 5.3 MongoDB

MongoDB is the datacommunication layer used to communicate between the MAS and Ros layer. MongoDB is a no-SQL database, which means that it operates with a flexible schema. From the mongo website: "*documents in the same collection do not need to have the same set of fields or structure, and common fields in a collections documents may hold different types of data.*"[7]

## 6 Literature

The majority of the literature will consist of the papers and thesis' that Daniel Telgen and Leo van Moergestel have written themselves. All the papers in this context are accepted at international conferences[8]. These papers are available on google drive.

[7]http://docs.mongodb.org/manual/core/data-modeling/
[8]Most of them. If they where rejected this will be noted.

# 7 Risks

| Severity | Chance | Risk | Solution |
|---|---|---|---|
| 10 | 3 | Because there are a lot of bugs to fix in REXOS it is possible there will not be enough time left for the actual research | This can be avoided by keeping track of upcoming deadlines and prioritising bugs that can be fixed in the time alotted |
| 3 | 6 | When running performance tests on a system that is not yet fully stable, it is possible faulty data will be gathered and the results will be inconsistent | To limit this, it is useful to know about every aspect in REXOS so a faulty process can be identified earlier |
| 5 | 2 | Failing server with loss of documents / wiki entries | Keep an off-site backup of all the data and having a backup server available |
| 8 | 4 | Not being able to get sensible conclusions out of the performance test results | Make sure that the results are reproducable and the test data that is being gathered is sensible. Also keep an eye on what we want to accomplish with gathered data. If the gathered data has no meaning afterwards, we dont need it. |
| 5 | 3 | Having an unstable JADE platform | When stabilizing the REXOS platform, keep testing with different data and different cases. While improving the platform, keep an eye on how JADE responds to it and this should not happen |
| 5 | 3 | MongoDB not being the right engine for blackboards on a large scaled platform, i.e. it is too slow | When this happens, there will be two options. One is to look into the problem if the performance can be improved. The other is to look into alternatives of MongoDB. |
| 10 | 1 | ROS not being able to perform on a large scaled platform | When this happens, the project has to be redesigned, or a different machine control environment has to be chosen. But it is very unlikely this will happen since ROS is designed to be scalable[11]. |

Table 2.3: Project risks

# The assignment

## 1 Problem description

Project REXOS is a fairly young project[1], meaning that alot of the architecture and structure are not properly implemented yet. The first problem to tackle will be to significantly re-organise and properly restructure the current code / project. It is also important that the project is able to deal with multiple product agents and multiple equiplets.

The previous implementation of the equiplet[2] has only a few things in common with the current implementation. The current implementation of the equiplet fits better into the architecture. Due to the changes that were made, the previous performance test results are no longer accurate and therefore it is unknown what the performance of the current implementation actually is. Also, the previous performance tests lacked in depth information about the performance of the REXOS platform.

In order to make sure that the current architecture is feasible for multiple equiplets, the performance has to be retested. Testcases have to be made for the different aspects of REXOS, for example: message communication of the JADE part, chatter of ROS nodes and IO speeds of the Blackboards.

Reconfigurability is one of the key features needed in an RMS such as REXOS.[3] Due to certain other priorities withing the project, this subject has not been researched / implemented yet. Because of the importance of reconfigurability, this will be the main problem during the semester.

## 2 Goals

### 2.1 Stable REXOS platform

Prior to the research that has to be done. The current platform's stability has to be improved. This has to be done because testing solutions or performing tests will be more accurate and reliable. Also when a stable platform is provided to a next project, developing or researching using that platform will be easier and more reliable. Stabilizing a platform can be done continuously, but in this project it will have a time limit and when the project supervisor thinks that the platform is stable enough.

[1]Started in 2008

[2]It was built for demonstrating purposes

## 2.2 Performance

After the platform has been stabilized, the platform will be tested. The goal of testing the platform is to get a picture on how 'good' the current platform is. A conclusion can be made out of the performance results if certain parts of the REXOS architecture has to be changed or improved. An important feature that has to be reached of REXOS is scalability. Untill today there has not been any research to the scalability of REXOS. With the performance results, the scalability of the current REXOS platform can be described.

## 2.3 Reconfigurability

Reconfiguring an equiplet is one of the main features of a future REXOS platform in a factory. Up until this moment there is not yet any research of implementing reconfigurability of an equiplet within the REXOS platform. With the research to reconfigurability within the REXOS platform, the goal is to create a proposition of how reconfigurability can be implemented on the MAS level of REXOS. The document will describe what has to be changed within the REXOS architecture in order to make reconfigurability work on a higher level.

In short the goals of this project are:

- gaining a stable platform to have reliable research results and that future projects can continue without big issues
- Getting a picture of how the current platform performs and if the platform needs improvement or change
- Create a proposition on how to implement reconfigurability within the REXOS platform, so one of the key features of a future REXOS can be developed.

# 3 Research assignment

Because our research is only limited to certain aspects of the REXOS project, we had to establish clear project boundaries. Our project assignment consist of one main task and several smaller sub-tasks.

## 3.1 Main assignment

The main assignment is to research the best way of implementing reconfiguration, and when time allows for it, implement it as well.

In this assignment, a list of pro's and con's will be compiled for different ways of implementing reconfigurability, the possibilities have not been researched as of yet. One this list is done, the team will make a decision about how to implement reconfigurability.

Once a decision has been made, code will need to be written for the ROS components as well as for MAS. This code will need to be tested thoroughly to make sure it meets the quality standard.

If for any reason, the implementation revealed difficulties or in some way proved to be impossible, another method of implementation can be chosen from the initial list of options. This does mean however, that new code will need to be written and tested.

A report will be compiled with all of the implementation options, their pro's and con's, the reasons for choosing the implemented method and any difficulties encountered along the way.

While working on this assignment, there are the following risks:

- It might prove necessary to modify large amounts of the base code
- An efficient method of implementing reconfigurability might not exist

## 3.2 Sub assignments

The subtasks in this project are defined to be the stabilisation of the platform and the testing of performance. This means that we need to gather a broad range of metrics to ascertain whether or not the system can cope with various loads. There currently a number of known sub assignments, but these are subject to change throughout the project.

The known sub assignments are:

- Stabilizing the platform

  The platform is currently not running in a stable manner, there are inconsistencies in how jade agents communicate with each other and with ROS, this creates confusion during debugging. This is just one result from having the previous implementation of the platform written as a demonstrator.

- Implementing multiple product agents and equiplets

  Because the platform was set up as a demonstrator, multiple product agent support and multiple equiplet support has not been implemented. This is required for collecting meaningful test data.

- Demonstrators The equiplets in the lab will be displayed at the precision fair[3] in 2013. For this reason demonstrators need to be written and tested extensively.

  The demonstrators going on display will feature the following equiplets:

  - 3D-Printer
  - Pick-and-place
  - Stewart-Gough

- Performance tests

  The performance of the platform is unknown, for this reason tests must be run and their data well documented. This data will be used in an extension of the FAIM[5] Paper in a journal.

- Infrastructure of the project

  The current filestructure is quite messy and requires cleaning to meet the quality criteria. The source code requires refactoring to comply to the new code standard.

## 3.3 Project boundaries

Since the project is of a rather large size, the project boundaries need to be well defined. This means that most aspects of the project our out of scope.

- Within Scope
  - Creating and maintaining the software related to ROS and MAS.
- Out of Scope
  - Any aspects concerning the hardware of the equiplets. The vision related aspects

[3]http://www.precisiebeurs.nl

# 4 Research topics

According to the problem description, the following research topics will be answered at the end of this research project.

- *Is the current hybrid architecture of REXOS with ROS, Blackboard, MongoDB and Jade usable for a grid of (arbitrary)100 equiplets?*
  - Is the current architecture scalable?
  - When is the performance high enough?
  - How can we test if the performance is high enough?
  - What can be improved in the architecture?
- *How far can an equiplet or parts of an equiplet be reconfigured?*
  - Which steps can be fully automated and which steps have to be done manually?
  - What consequences, if any, does this have for the architecture?
  - Which parameters require human input?
  - How do you add new functionality to a grid?
    * Which demands must a reconfigurable equiplet satisfy?
    * How do we prove that the grid is the best manufacturing layout?
    * For what purposes is the concept applicable?

# 5 Means of research

## 5.1 Performance REXOS

For this part quantitative research will be used. This is the gathering of statistics, which will then be used to create graphs and diagrams. These will then be used to assess the performance of the REXOS platform.

To gather and process these statistics tools will need to be created. It was decided to create a logger which logs all messages between agents, between ROS nodes and between the two previous and blackboards. Besides this logger, a parser to read the logs will also be created. This parser will be capable of turning the log entries into metrics which can then be used to generate statistics.

Taking into consideration that there are not very many equiplets to test with, a virtual equiplet will need to be created in the software, this equiplet will need to run everything a regular equiplet does, wih the exception of its modules, which are after all virtual. This also means virtual modules will need to be created.

## 5.2 Reconfigurability

To determine which way to implement reconfigurability, the possiblities will need to be researched and using a pro/con table it will be possible to make a number of initial choices. Based on these initial choices, research can be done to see what impact they would have on the current architecture. Finally, a decision will be made as to how it will be implemented.

To implement reconfigurability, additional software needs to be written and added to the platform. Tests will need to be run every step of the way, to ensure the correct functioning of the platform. These tests include the following:

- Running through the entire process of creating a product, this process includes the following actions
  - Spawning agents
  - Entering product steps
  - Translating product steps
  - Scheduling equiplets
  - Executing steps
- Calibrating modules, such as
  - Delta robot
  - Camera
- Connecting modules
- Disconnecting modules

Every step during this part of the research will be documented and the end results will be delivered to the project supervisor. For a full list of products please check section 6

# 6 Final products

At the end and during this research project, a variety of documents has to be made to make sure the project can be handed in so a follow-up project can start without ambiguities. A thesis has to be handed in containing the following items. These items apply for both main research topics.

- An analysis of the problems described
- describing possible solutions for the problems
- An argumented recommendation/choice of the proposed solutions
- evaluations of the made recommendations/choices and
- an evaluation of the whole result according to the assignment

More items can be added when needed. Both research topics will have some final products of its own; and will be described in the following 2 sections.

## 6.1 Reconfigurability

At some time, the REXOS platform has to be able to adapt when an equiplet has been reconfigured i.e. another module has been attached to the equiplet. This has to be done by writing additional software to REXOS. Research needs to be conducted to define the possible solutions of REXOS being able to handle reconfigurable equiplets. Test software will be developed to get more information on what the best solution is for REXOS. The most important code will be documented and delivered. The conclusion on what the best implementation of reconfigurability within REXOS will be described in the thesis.

## 6.2 Performance testing

Results of performance testing and follow-up recommendations according to the analyzed test data will be described in the thesis. Testing performance in a platform means that there has to be software written to gather data and analyze it. The written software will be documented and delivered.

# 7 Quality criteria

In order to deliver a proper reasearch result, control of quality needs to be defined. This means that certain quality criteria should be established to which the final result shall be hold up against. Those criteria are:

- *The results are measurable*
  Due to the nature of the project, some final results won't be measurable. The results that are will be will be measured with the research questions in mind, determining whether we resloved the question.
- *The results are well documented*
  It is important that all of our research is documented. Due to the state of which the project currently is in, we should also consider documenting existing software from the project. Documenting the research means writing down choices made, outcomes of possible research and recording every action where a (difficult) choice was made.

If one of the results fail to hold up against these criteria, it should be deemed unusable as a proper result.

# References


[1] M. Wooldridge & N. Jennings, *Intelligent agents: Theory and practice*. The Knowledge Engineering Review, (10(2)):115152, 1995.

[2] Zhang Z. & McCalley J. D., *Multiagent System Solutions for Distributed Computing, Communications, and Data Integration Needs in the Power Industry*

[3] Y. Koren et al, *Reconfigurable Manufacturing Systems*, CIRP Annals- Manufacturing Technology, vol. 48 no. 2, page: 527-540, 1999.

[4] E. Puik and L.J.M. van Moergestel, *Agile multiparallel micro manufacturing using a grid of Equiplets*, IPAS 2010 proceedings, page: 271-282, 2010.

[5] D. Telgen et al, *Machine States for Distributed Control Systems for Reconfigurable Manufacturing Systems*, 2013

[6] A. Gunasekaran, *Agile manufacturing: the 21st century competitive strategy*, Elsevier, 2001.

[7] Leo van Moergestel, Erik Puik and Daniel Telgen, *Production scheduling in an agile agent-based production grid*, Web Intelligence and Intelligent Agent Technology (WI-IAT), 2012 IEEE/WIC/ACM International Conferences Volume:2 Page(s): 293 - 298, 2012.

[8] D. Telgen, L. van Moergestel, E. Puik, A. van Zanten, and A. Abdulamir *Automatic Structured Decomposition of Manufacturing Actions in an Agent-Based Manufacturing System* , IAT 2013

[9] No Author Given, No Institute Given *Combining Performance and Flexibility for RMS with a Hybrid Architecture*

[10] Leo van Moergestel, Erik Puik and Daniel Telgen, *Production scheduling in an agile agent-based production grid*, Web Intelligence and Intelligent Agent Technology (WI-IAT), 2012 IEEE/WIC/ACM International Conferences Volume:2 Page(s): 293 - 298, 2012.

[11] Morgan Quigley et al, *ROS: an open-source Robot Operating System*, Proc. Open-Source Software workshop of the International Conference on Robotics and Automation (ICRA), 2009.

[12] Stewart, D.A, *Platform with six degrees of freedom*, Proc. Inst. Mech. Engr, Vol.180, Part I, No. 15, 1965/1966, pp. 371- 386.

[13] K. Schwaber, M. Beedle, *Agile Software Development with Scrum.*