RobMoSys

H2020—ICT—732410

# RobMoSys

## Composable Models and Software for Robotics Systems

## Deliverable D6.3: Eclipse Project and Eclipse Project Proposal

Gaël Blondelle (EFE)
Dennis Stampfer (HSU)

This project has received funding from the *European Union's Horizon 2020 research and innovation programme* under grant agreement No. 732410

RobMoSys

*Project acronym*: RobMoSys

*Project full title*: Composable Models and Software for Robotics Systems

*Work Package*: WP 6 - Exploitation

*Document number*: D6.3

*Document title*: Eclipse Project and Eclipse Project Proposal

*Version*: 1.0

*Due date:* December 31th, 2017

*Delivery date*: 22.12.2017

*Nature*: Report (R)

*Dissemination level*: Public (PU)

*Editor:* Gaël Blondelle (EFE), Dennis Stampfer (HSU)

*Author(s)*: Gaël Blondelle (EFE), Philippe Krief (EFE), Dennis Stampfer (HSU), Alex Lotz (HSU), Christian Schlegel (HSU), Matteo Morelli (CEA), Enea Scioni(KUL)

*Reviewer:* Susanne Bieller (EUR)

# Executive Summary

RobMoSys is about introducing model-driven software engineering to robotics. Thus, RobMoSys has to manage the interfaces between different roles (e.g. robotics expert, domain expert, component supplier, system builder, installation, deployment and operation) and separate concerns in an efficient and systematic way by making the step change to a set of fully model-driven methods and tools for composition-oriented engineering of robotics systems.

This document represents the first annual deliverable of D6.3 “Eclipse Project and Eclipse Project Proposal”. The document reports on the progress and plans of creating an Eclipse project to publish and promote the results of RobMoSys in order to push its acceptance to a wider audience of developers and adopters by hosting it by the Eclipse Foundation.

The deliverable paves the path of RobMoSys publishing Eclipse projects. It therefore first describes how Eclipse projects are established. The document then presents a generic mapping of the way in which RobMoSys organizes its ecosystem to the Eclipse ecosystem and to multiple Eclipse projects. One of the conclusions of this document is that a single Eclipse project representing the results of RobMoSys is not adequate.

Concrete activities within RobMoSys are identified as potential candidates for Eclipse projects. First activities within RobMoSys have started to draft an Eclipse project proposal.

# Content

# 1 Introduction

RobMoSys is about introducing model-driven software engineering to robotics. Thus, RobMoSys has to manage the interfaces between different roles (e.g. robotics expert, domain expert, component supplier, system builder, installation, deployment and operation) and separate concerns in an efficient and systematic way by making the step change to a set of fully model-driven methods and tools for composition-oriented engineering of robotics systems.

This document represents the first annual deliverable of D6.3 “Eclipse Project and Eclipse Project Proposal”. The document reports on the progress and plans of creating an Eclipse project for the results of RobMoSys in order to push the acceptance and incubation by hosting it by the Eclipse Foundation.

Originally, the RobMoSys consortium planned to create one single Eclipse project for all RobMoSys results. However, it turns out that in the way RobMoSys is organizing its ecosystem, a mapping of individual “parts” of RobMoSys to individual Eclipse projects is more adequate. This document serves as a self-analysis of the RobMoSys project to identify a potential mapping of RobMoSys to the Eclipse ecosystem. It identifies concrete project candidates and reports on the progress of establishing them as Eclipse projects.

The document is structured as follows: First, it explains the Eclipse ecosystem in general. Then it elaborates how open source projects work at Eclipse in particular. Finally, the document connects to RobMoSys and describes how RobMoSys can benefit from the Eclipse ecosystem in order to establish and grow its own open source community and report on its progress.

This deliverable will be updated on an annual basis. It is related to Deliverable D7.2 “Business Models for the Ecosystem” (RobMoSys Consortium 2017a) and D7.3 “Sustainability Plan” (RobMoSys Consortium 2017b) that describes the strategy for RobMoSys sustainability including community governance.

# 2 The Eclipse Ecosystem

Deciding to contribute to open source, as a consumer, as a producer of open source components, or both, is a volunteer act to jointly manage some software code and all the resources attached to this software. Deliverable 7.2 (RobMoSys Consortium 2017a) describes open source ecosystems in general. In this deliverable, we focus on the Eclipse ecosystem in particular as we expect to use this ecosystem as a channel to address a larger developper audience.

The Eclipse community gathers individuals and organizations that wish to collaborate on business-friendly open source software. Its projects are focused on building an open development platform comprised of extensible frameworks, tools and runtimes for building, deploying and managing software across the lifecycle. The Eclipse Foundation is a not-for-profit, member-supported corporation that hosts the Eclipse projects and helps cultivate both an open source community and an ecosystem of complementary products and services.

The Eclipse Project was originally created by IBM, in November 2001, and was supported by a consortium of software vendors. Industry leaders like Borland, IBM, MERANT, QNX Software Systems, Red Hat, SuSE, TogetherSoft, and Webgain formed the initial eclipse.org Board of Stewards. By the end of 2003, this initial consortium had grown to over 80 members. In 2017, there are over 260 Eclipse members.

The Eclipse Foundation, Inc. was created in January 2004 as an independent not-for-profit corporation to act as the steward of the Eclipse community. It was created to allow a vendor neutral, open and transparent community to be established around the Eclipse projects. At the end of 2017, the community consists of individuals and organizations from a cross section of the software industry. All technology and source code provided to and developed by this fast-growing community is made available royalty-free via the Eclipse Public License (EPL).

From its inception, the Eclipse Foundation has been designed to be a business friendly ecosystem (Fig. 1). It not only fosters collaboration on software platforms by the obligation of publishing code modification(s) when redistributing software licensed under the EPL. But is also allows the usage of Eclipse projects in any context, including as the framework of proprietary products. The latter is due to the permission of relicensing of Eclipse projects binaries under any license when they are extended to create products.

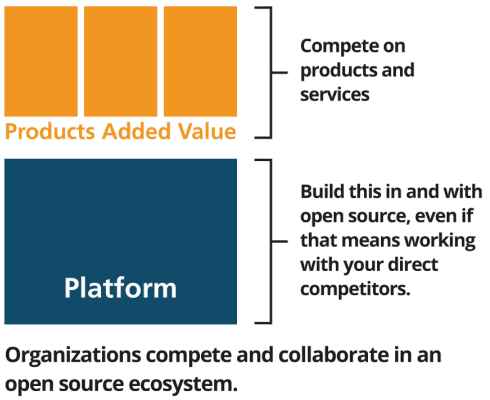


**Figure 1**: A business friendly ecosystem based on extensible platforms

## 2.1 The Eclipse Developer Community

In 2017, the Eclipse community consists of:

- Over 330 open source projects;
- 130 million lines of code delivered every year;
- Over 1.400 individual committers;

- Over 5 million of active users;
- An average of 1,5 million downloads per month;
- An average of 2 million unique website visitors per month;
- A leading IDE on Java, C/C++, PHP, etc.;
- More than 260 members (including 12 strategic members);
- More than 50 events organized or co-organized each year, like EclipseCon events, Eclipse Days, Eclipse Summits or Eclipse Demo Camps, etc.

Joining this community will improve the visibility of the RobMoSys project, the sustainability of the technologies after the end of the funded period, and will help with the best practices of open source.

## 2.2 Eclipse Working Groups

Companies looking to drive innovation and efficiencies within their own organizations are increasingly looking for external sources to advance new ideas. Commonly referred to as "open innovation," this paradigm encourages collaboration across organizational boundaries, and is often practiced in the open source community. In 2009, the Eclipse Foundation launched the notion of Working Groups to allow organizations to combine the best practices of open source development with a set of services required for open innovation, and in turn enabling organizations to foster industry collaborations.

Eclipse Working Groups provide a vendor-neutral governance structure that allows organizations to freely collaborate on new technology development. The Eclipse Foundation, through Eclipse Working Groups, provides five basic services to enable these types of collaborations (Fig. 2):


**Figure 2:** Pillars of open collaborations

- **Governance**: Good governance that controls how decisions are made, policies established and disputes resolved is important for any successful collaboration.
- **Intellectual Property Management and Licensing**: Collaboration among different organizations requires due diligences on the co-developed intellectual property. Eclipse Working Groups are established under the intellectual property (IP) policies of the Eclipse Foundation. These policies ensure that any open source software created in Eclipse

projects is available for use by anyone, including developers of commercial software products.

- **Development Processes**: The Eclipse community has created a successful development process for large-scale distributed development that involves many different organizations.
- **IT Infrastructure**: The Eclipse Foundation manages the IT infrastructure for Eclipse Working Groups, including Git code repositories, Bugzilla databases, Hudson CI servers, development-oriented mailing lists and newsgroups, download sites, and websites.
- **Ecosystem Development**: An important way that the Eclipse Foundation supports the community is through active marketing and promotion of Eclipse Working Groups and the wider Eclipse ecosystem.

At the end of 2017, the Eclipse Foundation manages several active working groups in various domains including:

- **openMDM**: The openMDM Working Group (http://www.openmdm.org/) wants to foster and support an open and innovative eco-system providing tools and systems, qualification kits and adapters for standardized and vendor independent **management of measurement data** in accordance with the ASAM ODS standard.
- **Internet of Things**: The IoT Working Group (http://iot.eclipse.org) is the place to learn about Eclipse technologies developed to make Internet of Things (IoT) development simpler.
- **LocationTech**: The LocationTech Working Group (https://www.locationtech.org/) is focusing on open source geospatial technologies.
- **Science**: The Science Working Group (http://science.eclipse.org/) is a collaboration of scientists developing software components used for basic **scientific research**.
- **PolarSys**: The PolarSys Working Group (https://www.polarsys.org/) was created by large industry players and by tool providers to collaborate on the creation and support of Open Source tools for the **development of embedded systems**.

# 3 The Eclipse Development Process

Over the years, the Eclipse community has created a successful process for large-scale distributed development that involves many different organizations. By publishing software in the Eclipse ecosystem, the RobMoSys project has to follow this process.

In this section, we summarize some of the principles driving the Eclipse Development Process (https://www.eclipse.org/projects/dev_process/).

## 3.1 Project Structure and Organization

A project is the main operational unit at the Eclipse Foundation. Specifically, all the open source software development occurs within the context of a project. Projects have leaders, developers, code, builds, downloads, websites, and more. Projects are more than just the sum of their many parts; they are the means by which open source work is organized when presented to the communities of developers, adopters, and users. Projects provide structure that helps developers expose their hard work to a broad audience of adopters, extenders, or end-users.

## 3.2 Committers

Each project has exactly one set of committers, the people who have right access to the project resources (code repository, documentation, ...). Each project's set of committers is distinct from

that of any other project. All project committers have equal rights and responsibilities within the project. Partitioning of responsibility within a project is managed using social convention.

The committers of a project have the exclusive right to elect new committers to their project; no other group can force a project to accept a new committer.

## 3.3 Code and Resources

Each project owns and maintains a collection of resources. These resources may include source code, a project website, space on the download servers, access to build resources, and other services provided by the Eclipse Foundation infrastructure. The exact infrastructure provided by the Eclipse Foundation varies over time and is defined outside this document.

Namespaces are assigned to a project by the Eclipse Management Organization (EMO). All project source code must be organized in the assigned namespaces and projects can only release code under their own namespace (that is, they cannot infringe on another Eclipse project's namespace).

## 3.4 Leaders

There are two different types of Eclipse project leadership: the Project Management Committee (PMC) and Project leaders. Both forms of leadership are required to:

- ensure that their project is operating effectively by guiding the overall direction and by removing obstacles, solving problems, and resolving conflicts;
- operate using open source rules of engagement: meritocracy, transparency, and open participation; and
- ensure that the project conforms to the Eclipse Foundation IP policy and procedures.

## 3.5 Committers and Contributors

Each project has a development team, led by the project leaders. The development team is composed of committers and contributors. Contributors are individuals who contribute code, fixes, tests, documentation, or other work that is part of the project. Committers have write access to the project's resources (source code repository, bug tracking system, website, build server, downloads, etc.) and are expected to influence the project's development.

Contributors who have the trust of the project's committers can, through election, be promoted committers for that project. The breadth of a committer's influence corresponds to the breadth of their contribution. A development team's contributors and committers may (and should) come from a diverse set of organizations. A committer gains voting rights allowing them to affect the future of the project. Becoming a committer is a privilege that is earned by contributing and showing discipline and good judgment. It is a responsibility that should be neither given nor taken lightly, nor is it a right based on employment by an Eclipse member company or any company employing existing committers.

The election process begins with an existing committer on the same project nominating the contributor. The project's committers will vote for a period of no less than one week of standard business days. If there are at least three positive votes and no negative votes within the voting period, the contributor is recommended to the project's PMC for commit privileges. If there are three or fewer committers on the project, a unanimous positive vote of all committers is substituted. If the PMC approves, and the contributor signs the appropriate committer legal agreements established by the Eclipse Management Organization (EMO) (wherein, at the very least, the developer agrees to abide by the Eclipse Intellectual Property Policy), the contributor

becomes a committer and is given write access to the source code for that project.

At times, committers may become inactive for a variety of reasons. The decision-making process of the project relies on active committers who respond to discussions and vote in a constructive and timely manner. The project leaders are responsible for ensuring the smooth operation of the project. A committer who is disruptive, does not participate actively, or has been inactive for an extended period may have his or her commit status revoked by the project leaders. Unless otherwise specified, "an extended period" is defined as "no activity for more than six months".

## 3.6 Project Plans

Projects are required to make a project plan available to their community at the beginning of the development cycle for each major and minor release. The plan may be as simple as a short description and a list of issues, or more detailed and complex.

Project Plans must be delivered to the community through communication channels approved by the EMO. The exact nature of the project plan varies depending on numerous variables, including the size and expectations of the communities, and requirements specified by the PMC.

## 3.7 Mentors

New project proposals are required to have at least one mentor. Mentors must be members of the Architecture Council. The mentors must be listed in the proposal. Mentors are required to monitor and advise the new project during its incubation phase; they are released from that duty once the project graduates to the mature phase.

## 3.8 Development Process

As indicated on the following figure, projects go through distinct phases (Fig. 3). The transitions from phase to phase are open and transparent public reviews.

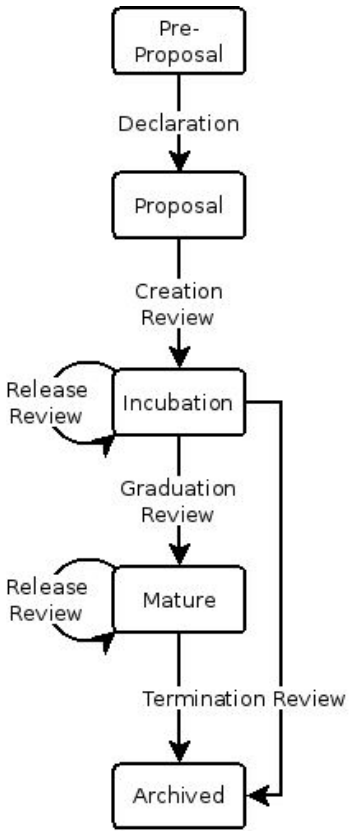


**Figure 3:** Eclipse Development Process

***Pre-proposal Phase***

An individual or group of individuals declares their interest in, and rationale for, establishing a project. The EMO will assist such groups in the preparation of a project proposal. The pre-proposal phase ends when the proposal is published by EMO and announced to the membership by the EMO.

***Proposal Phase***

The proposers, in conjunction with the destination PMC and the community, collaborate in public to enhance, refine, and clarify the proposal. Mentors for the project must be identified during this phase. The proposal phase ends with a creation review, or withdrawal. The proposal may be withdrawn by the proposers at any point before the start of a creation review. The EMO will withdraw a proposal that has been inactive for more than six months.

***Incubation Phase***

The purpose of the incubation phase is to establish a fully-functioning open-source project. In this context, incubation is about developing the process, the community, and the technology. Incubation is a phase rather than a place: new projects may be incubated under any existing project. A project in the incubation phase can (and should) make releases and the incubation phase ends with a graduation review or a termination review.

***Mature Phase***

The project team has demonstrated that they are an open-source project with an open and transparent process, an actively involved and growing community, and Eclipse-quality technology. The project is now a mature member of the Eclipse community. Major releases continue to go through release reviews.

***Archived***

Projects that become inactive, either through dwindling resources or by reaching their natural conclusion, are archived. Projects are moved to archived status through a termination review. If there is sufficient community interest in reactivating an archived project, the project can start again with a creation review. Since there must be good reasons to have terminated a project, the creation review provides a sufficiently high bar to prove that those reasons are no longer valid.

## 3.9 The Eclipse IP Process

Eclipse projects are expected to take necessary precautions to mitigate intellectual property (IP) risk to adopters. The Eclipse Intellectual Property Policy[1], and the process that implements it, are important so that companies wanting to integrate the open source code into their project can do so knowing that the project can legally be distributed under the agreed-to terms. The IP Due Diligence Process[2], managed by the Eclipse IP Team (commonly referred to as the IP Team), is in place to support this.

Code provenance tracking is critical; the Eclipse Foundation needs to know the source of all code that ends up in its repositories. To that end, all new projects are required to make an **initial contribution** before any code is committed to a project's source code repository.

This code originates from three sources at Eclipse:

1. Eclipse Committers
2. Community Contributions (typically in the form of bug fixes)
3. Content developed and maintained somewhere other than Eclipse (other open source projects)

The first two cases are addressed by asking committers and contributors to sign agreements before they contribute code and by making sure that the code has not been copied inappropriately, that licenses are being used correctly, and so forth.

The third case is the most complex one as it implies that the Eclipse Foundation checks all dependencies used in a project, as well as dependencies of dependencies, in order to make sure that proper IP management process is applied to these libraries.

Projects are not able to make a release until the due diligence on the IP contained in that release—including project code contributions and third-party libraries—is complete.

This IP process applies also to projects hosted by Eclipse Working Groups, so it applies to OpenCert as part of PolarSys.

The Eclipse Foundation uses a special tracker called IPZilla in order to manage intellectual property due diligences according the Eclipse IP process.

## 3.10 Proposal

Any project that wants to join the Eclipse Foundation has to submit a proposal. The proposal has to follow the guidance described in the following guidance document:

https://wiki.eclipse.org/Development_Resources/HOWTO/Pre-Proposal_Phase.

To summarize, the proposal has to define:

- **Project Name**: Naming and branding are challenging issues. In order to provide a consistent brand for Eclipse, projects must follow the project guidelines (https://wiki.eclipse.org/Development_Resources/HOWTO/Project_Naming_Policy).
  As a defensive measure, the Eclipse Foundation holds the trademark to the Project names on behalf of the Projects - this prevents companies from misusing or misrepresenting their

[1] https://www.eclipse.org/org/documents/Eclipse_IP_Policy.pdf
[2] https://www.eclipse.org/legal/EclipseLegalProcessPoster.pdf

products as being the Projects. The EMO will initiate a trademark review prior to scheduling a Creation Review. Existing trademarks must be transferred to the Eclipse Foundation (please see the Trademark Transfer Agreement[3]).

- **Project description**: the description must be clear, concise and understandable. It must use plain non-technical English. It must describe all acronyms. The project description should include the following sections:
  - Background: Describe where the project came from. What is the historical journey of the project; who/what company wrote the project? Did it go through any significant alterations/rewrites/language changes?
  - Scope: Provide an introductory paragraph describing what the project aims to be, followed by several bullet points. The scope should allow for some flexibility, but still provide well-defined boundaries for the project.
  - Description: The introductory paragraph should clearly explain what the project is and does. Think of this as an expanded elevator pitch.
  - Why here:
    - Why does this project want to be hosted at the Eclipse Foundation?
    - What do you expect to gain by having your project at the Eclipse Foundation?
    - What value does the project provide to the Eclipse community and ecosystem?
    - **Licenses**: Check the licenses that apply to the project.
    - **Legal Issues**: Describe any legal issues around the project and/or code.
    - List the current licenses of the main code.
    - List the 3rd party dependencies and associated licenses.
- **Initial Contribution**: Describe the initial contribution. Where is the code coming from? Current Eclipse project/GitHub repository or other.
- **Future Work**: How is the project going to grow its community (users / adopters / committers)?
- **Source Code** section:
  - Repository Source Type
  - Source Repositories
- **People**:
  - A project needs a project lead
  - Initial Committers
  - Mentor(s)
  - Interested Parties: Who is interested in this project? This could be individuals or companies.

## 3.11 Reviews

The Eclipse Development Process is predicated on open and transparent behaviour. All major changes to Eclipse projects must be announced and reviewed by the membership-at-large. Major changes include the project phase transitions as well as the introduction or exclusion of significant new technology or capability. It is a clear requirement that members who are monitoring the appropriate media channels not be surprised by the post-facto actions of the projects.

Projects are responsible for initiating the appropriate reviews. If it is determined to be necessary, the project leadership chain (e.g. the PMC or EMO) may initiate a review on the project's behalf.

[3] https://www.eclipse.org/legal/Trademark_Transfer_Agreement.pdf

## 3.12 Creation Review

The purpose of the creation review is to assess the community and membership response to the proposal, to verify that appropriate resources are available for the project to achieve its plan, and to serve as a committer election for the project's initial committers. The Eclipse Foundation strives not to be a repository of "code dumps" and thus projects must be sufficiently staffed for forward progress.

## 3.13 Graduation Review

The purpose of the graduation review is to mark a project's change from the incubation phase to the mature phase. The graduation review confirms that the project is/has:

- A working and demonstrable code base of sufficiently high quality.
- Active and sufficiently diverse communities appropriate to the size of the graduating code base: adopters, developers, and users.
- Operating fully in the open following the principles and purposes of the Eclipse Foundation.
- A credit to the Eclipse Foundation and is functioning well within the larger Eclipse community.

A graduation review is generally combined with a release review.

## 3.14 Release Review

The purposes of a release review are: (1) to summarize the accomplishments of the release, (2) to verify that the IP Policy has been followed and all approvals have been received, (3) to highlight any remaining quality and/or architectural issues, and (4) to verify that the project is continuing to operate according to the principles and purposes of the Eclipse Foundation.

## 3.15 Termination Review

The purpose of a termination review is to provide a final opportunity for the committers and/or Eclipse membership to discuss the proposed archiving of a project. The desired outcome is to find sufficient evidence of renewed interest and resources in keeping the project active.

## 3.16 Releases

Any project may make a release. Releases are, by definition, anything that is distributed outside of the committers of a project. If users are being directed to download a build, then that build has been released. All projects and committers must obey the Eclipse Foundation requirements on approving any release. All official releases must have a successful release review before being made available for download.

# 4 RobMoSys as an Eclipse project

In this section, we present a potential perspective on how the RobMoSys project can publish some results in the Eclipse ecosystem, and how they may be published in the coming years. This section first describes the organization of the RobMoSys Ecosystem. The section then shifts to a general mapping of RobMoSys to Eclipse projects and reports on the current progress.

## 4.1 The RobMoSys Ecosystem

The RobMoSys Wiki describes the organization of the RobMoSys ecosystem in three tiers (Fig. 4, see https://robmosys.eu/wiki/general_principles:ecosystem:start). Tier 1 structures the ecosystem in general for robotics. It is shaped by the drivers of the ecosystem that define an overall composition structure which enables composition on lower tiers and which the lower tiers conform to. Tier 2 conforms to these foundations, structuring the particular domains within robotics and is shaped by the experts of these domains, for example, object recognition, manipulation, or SLAM. Tier 2 is shaped by representatives of the individual sub-domains in robotics. Tier 3 conforms to the domain-structures of Tier 2 to supply and to use content. Here are the most "users" of the ecosystem, for example component suppliers and system builders. The number of users and contributors is significantly larger than on the above tiers as everyone contributing or using a building block is located at this tier.

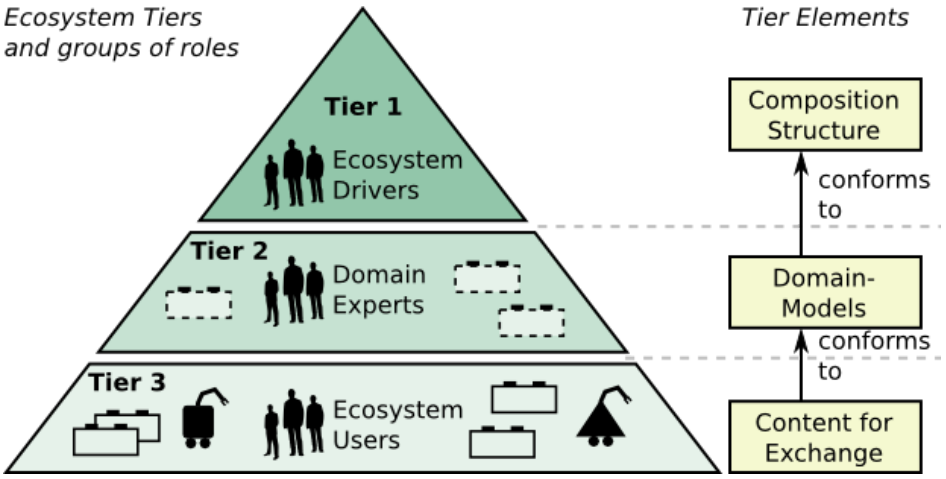


**Figure 4:** The RobMoSys Ecosystem Organization in three Tiers (see https://robmosys.eu/wiki/general_principles:ecosystem:start, figure from (Stampfer 2017)).

The RobMoSys ecosystem organization is compatible with the way the Eclipse ecosystem is organized (see Fig. 1). While RobMoSys Tier 1 introduces composition structures for robotics, the RobMoSys Tier 2 introduces structures for particular domains within robotics. Both the Tier 1 and Tier 2 map to the Eclipse "Platform" aspect in Fig. 1. As the Eclipse Ecosystem motivates, there is the need for openness and collaboration in RobMoSys Tier 1 and Tier 2 among the involved participants and representatives. In RobMoSys Tier 3, ecosystem participants provide and supply content, i.e. "they compete on products" (compare to "Products added value", Fig. 1).

The next section describes how the RobMoSys ecosystem organization in Tiers is suitable to map to individual Eclipse projects.

## 4.2 Granularity of an Eclipse project

In contrast to the original intention of RobMoSys to create one Eclipse project for RobMoSys, the RobMoSys consortium plans to create several Eclipse projects. This section describes the reasons behind this intention.

The RobMoSys project is about establishing structures to realize a step-change towards a European ecosystem for industry-grade software development. The RobMoSys results will include these structures but also supporting tools and pilot applications to demonstrate and evaluate the benefit of the RobMoSys approach. For these different kind of contributions by the RobMoSys consortium and the contributions of the open call projects, it is reasonable to separate the parts of the RobMoSys project into individual Eclipse projects. Otherwise, the different kind of contributions would amalgamate. In previous research projects, the Eclipse Foundation learnt that it is usually better to create projects with a smaller granularity rather than creating a single open source project to publish all the results. Indeed, every open source project has its own life cycle and having a smaller granularity encourages reuse of the technology in different contexts while providing a better visibility globally for the research project.

Structure, content that is conform to the structure, applications that demonstrate the benefit, and supporting tooling distinguishes in many aspects and are therefore candidate to be published in individual open source projects. For example they distinguish in the parties that drive the activities, in the level of abstraction and granularity, and with respect to governance (e.g. organizational and democratic structure).

RobMoSys therefore does not plan to create one Eclipse project, but maps to several tightly linked Eclipse projects at coarser granularity. The next section will map the elements of the RobMoSys project and comment on each one's suitability to become an Eclipse project.

## 4.3 Mapping of the RobMoSys Ecosystem to the Eclipse Ecosystem

The section "The RobMoSys Ecosystem" elaborated on the compatibility of the RobMoSys ecosystem with the Eclipse ecosystem. The similarity between the RobMoSys ecosystem Tiers and the Eclipse ecosystem organization motivates a direct mapping of the RobMoSys project to Eclipse as illustrated in Fig. 5.

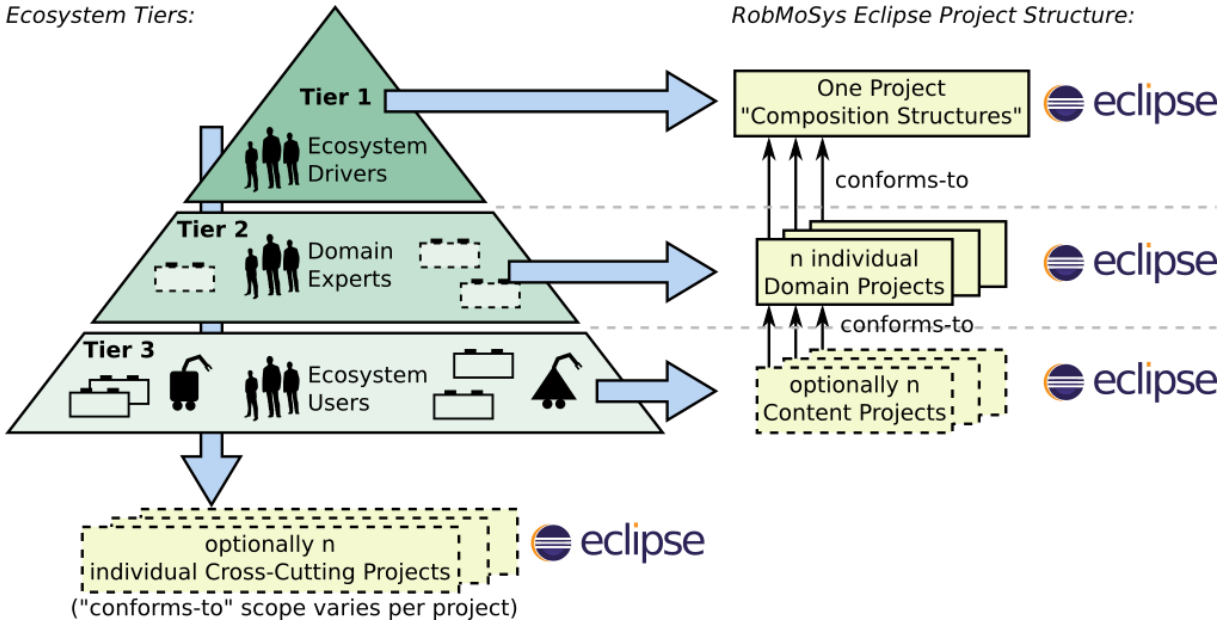


**Figure 5**: A generic concept of the RobMoSys Eclipse project structure. It is defined by the mapping of the RobMoSys ecosystem (left) to Eclipse projects (right).

Any lower RobMoSys ecosystem Tier depends on the upper Tiers, i.e. it directly conforms to its upper tier. In that sense, the individual Eclipse projects can and should be independent and self-organizing but must keep the strong link to the parent projects to maintain the conforms-to relation between the Eclipse projects that originate from the Tiers. This section will now address each kind of Eclipse project.

### 4.3.1 RobMoSys Tier 1: Composition Structures Project

- Content and Goal
  - This project is to contain the core composition structures of RobMoSys (see https://robmosys.eu/wiki/modeling:composition-structures:start). This project lies in the main responsibility of RobMoSys. The goal is to ensure broad visibility for the RobMoSys approach and to continue maintenance after the funding of the RobMoSys project ends.
  - Possible content, for example, are the composition structures that are expressed as technology-independent meta-models.
- Who runs this project
  - This is kick-started by RobMoSys consortium and extended by the RobMoSys consortium based on the results of the open call projects.
  - It is maintained by the RobMoSys consortium during the project run-time and must transition to community effort after the funding ends. An Eclipse project would provide the necessary attention for such a continuity. It may be driven by an Eclipse Working group (see section “RobMoSys as an Eclipse Working Group”).
- Target Group
  - The target group for this project is mainly the Eclipse projects that are based on Tier 2. The RobMoSys ecosystem users on Tier 3 are indirectly users of these structures. The Tier 1 project has a strong influence on cross-cutting projects that, for example, implement the RobMoSys structures to make them accessible by tooling.
- Suitability as an Eclipse project
  - Content within the RobMoSys Ecosystem Tier 1 **must be open**. Therefore, it is **very suitable as an eclipse project**.

### 4.3.2 RobMoSys Tier 2: Domain Structures as Individual Projects

- Content and goal
  - Several projects are possible on a per-domain granularity. Projects are to contain the domain-specific structures of RobMoSys that conform to the core composition structures (see https://robmosys.eu/wiki/general_principles:ecosystem:start).
  - Covering all robotics domains in Ecosystem Tier 2 is not the aim of RobMoSys. However, RobMoSys develops a concrete domain-structure for the “Motion, Perception, and World Model Stack”. This is one example for content of a Tier 2 project.
- Who runs the project
  - In long-term, individual projects should be established on a per-domain basis and run by representatives of the individual domains, keeping a close link to the Tier 1 project to maintain the conforms-to relation. In the short term, it may be desirable that RobMoSys kickstarts domain-structures in one single project, e.g. via the “Motion, Perception, and World Model Stack”.
- Target group
  - Users of the RobMoSys Ecosystem on Tier 3.
  - Tooling-providers in cross-cutting activities.
- Suitability as an Eclipse project
  - Content on RobMoSys Ecosystem Tier 2 strongly **should be open**. There is,

however, no must. Content on this tier is **very suitable as an Eclipse project**

### 4.3.3 RobMoSys Tier 3: Optional Eclipse projects on an Individual Basis

- Content and goal
  - On this ecosystem Tier, there are specific building blocks or software assets that are provided by suppliers and used both by system builders and other RobMoSys roles.
  - The granularity of Eclipse projects on this Tier very much depends on the kind of building blocks, on their scope, and on the potential business model of the supplier. There is a huge potential of an open source building block to actually become an Eclipse project. However, it is expected that there is a large number of proprietary building blocks on this Tier to provide a business perspective. Commercial building blocks will not become Eclipse projects.
  - Content on this Tier must keep the link to corresponding Tier 2 projects and indirectly to Tier 1 projects to maintain the conforms-to relation between the tiers.
  - Content on this Tier, for example, may be a a open-source robotics navigation solution developed and maintained by the community such as the publicly available open source components by HSU (see https://robmosys.eu/wiki/baseline:components:smartsoft).
- Who runs the project
  - On an individual basis depending on the scope and granularity of a building block.
  - An Eclipse project on this RobMoSys Tier is suitable to be run by any supplier role in RobMoSys (e.g. component supplier, behavior developer, system architect).
  - A project may be run by a whole community or by a company that feels responsible for the building block and, for example, provides commercial services around the building block.
- Target group
  - RobMoSys roles that are ecosystem users: e.g. system builder, component supplier
- Suitability as an Eclipse project
  - Contributions on RobMoSys Ecosystem Tier 3 **can be open**. Proprietary and commercial content will most probably exist. Therefore, content on this tier **can be an eclipse project**.

### 4.3.4 Cross Cutting Topics: Optional Eclipse projects on an Individual Basis

- Content and goal
  - There are several cross-cutting topics on the RobMoSys ecosystem that map across more than one RobMoSys Tier.
  - A very prominent example of a cross-cutting topic is tooling that is not bound to a particular building block or a concrete domain. Further, software cross-cutting software frameworks that do not focus on a particular tier can be found here.
  - In contrast to cross-cutting topics, there are particular frameworks or particular tooling for specific building blocks or RobMoSys Tiers. They should be part of projects covering the particular Tier or building block. For example: an Android app that supports in setting up path-based navigation on Tier 3 or a tool that supports in modeling kinematic constraints for the “Motion, Perception, World Model Stack” on Tier 3.
  - Content for cross-cutting projects, for example, are tooling and frameworks that cover the whole RobMoSys workflow and structures.
  - Cross-cutting projects have varying conforms-to relations, depending on the scope of the project.
- Who runs the project
  - Run on an individual basis, for example run by tooling providers.

- Target group
  - Varying depending on the project scope.
- Suitability as an Eclipse project
  - Suitability as an Eclipse project very much depends on the scope. For example, tooling **can be open source** and with/without commercial support. These projects **can be Eclipse projects**. The RobMoSys project establishes a basic set of tooling which is expected to be open source and therefore is a candidate to consider as an Eclipse project.

## 4.4 Analysis of RobMoSys activities with respect to potential Eclipse projects and Project Status

Based on the mapping of the RobMoSys ecosystem to Eclipse projects, this section describes potential project candidates as well as their plan and current status of becoming Eclipse projects.

### 4.4.1 Eclipse project for RobMoSys Tier 1

Tier 1 hosts the composition structures of RobMoSys. The main content are meta-models. Fig. 6 illustrates a potential realization for the RobMoSys Tier 1 as an Eclipse project.

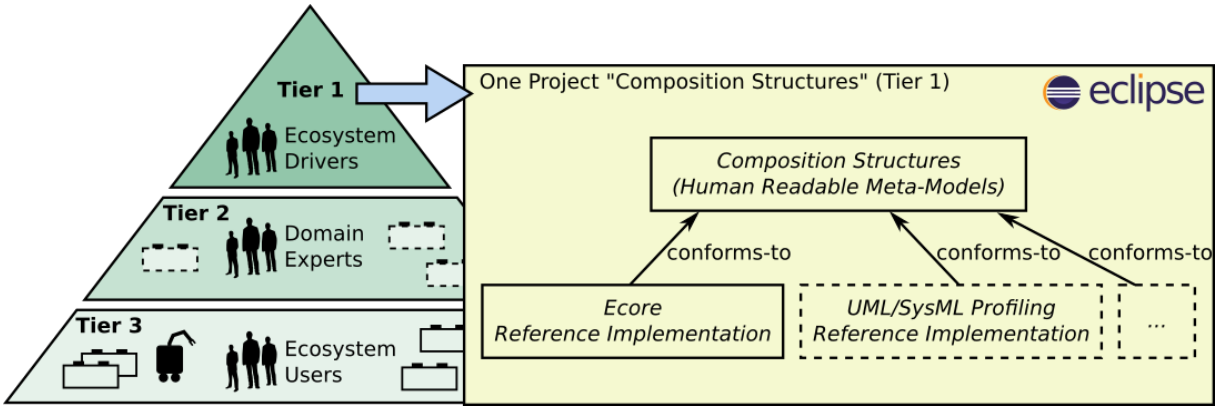


**Figure 6**: A potential realization of an Eclipse project to represent RobMoSys Tier 1.

At the moment, the consortium is working on the meta-models of the RobMoSys composition structures in a publicly accessible wiki (https://robmosys.eu/wiki/modeling:composition-structures:start). These meta-models are maintained in an abstract "human-readable" representation that provide the "first-class" representation of structures. RobMoSys by purpose does not provide these "first-class" meta-models in a specific implementation language such as Ecore or UML profile. This would add a language-specific and technological flavor to the composition structures that aimed to be generic. There are several alternatives on how the composition structures can be implemented. Examples can be found at https://robmosys.eu/wiki/modeling:realization_alternatives.

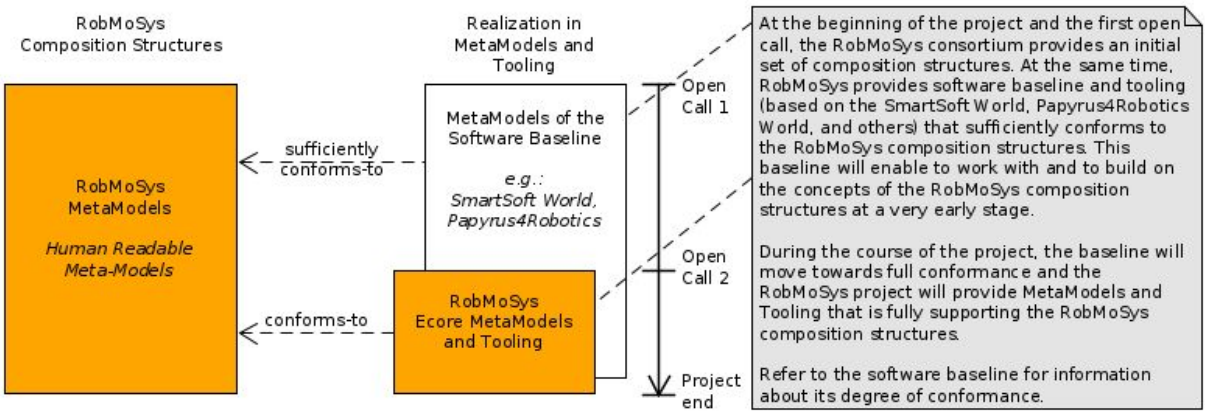


**Figure 7**: The RobMoSys modeling roadmap (figure from https://robmosys.eu/wiki/modeling:roadmap)

In order to actually apply the RobMoSys approach and to support users via tooling, the RobMoSys project makes available an Ecore-based realization of RobMoSys composition structures (Fig. 7). This Ecore realization plays the role of a pivot format enabling EMF-based software baselines to exchange models easily and effectively, instead of forcing the development of import/export facilities from/to each one of them. Of course, even tools built/running outside of the EMF ecosystem benefit from such a pivot format, which would be their entry point to the Eclipse world, e.g., to perform a specific analysis that only a given EMF baseline supports.

The project is currently working on an example implementation for these meta-models via Ecore (Fig. 7, see also https://robmosys.eu/wiki/modeling:roadmap). Both the human-readable meta-models and the potential implementations in various languages are candidates to become an Eclipse project. It makes sense to unite them in one Eclipse project to keep the strong link between the "human-readable" standard and the various reference implementations of the meta-models.

### 4.4.2 Eclipse projects for RobMoSys Tier 2

RobMoSys is currently working on domain models for "Motion, Perception, and World Model". RobMoSys will consider this as a candidate to become part of the envisioned Eclipse projects ecosystem.

### 4.4.3 Eclipse projects for RobMoSys Tier 3

Currently, most of the readily usable building blocks for RobMoSys on Tier 3 exist within the SmartSoft World which is conform to the RobMoSys structures. There are about 36 software components publicly available for immediate composition through the SmartMDSD Toolchain (see https://robmosys.eu/wiki/baseline:components:smartsoft). In principle, many of the individual components (e.g. obstacle avoidance component for wheeled robots) or even complete stacks of components (components realizing the flexible navigation stack for wheeled robots) are suitable to become an Eclipse project. However, there are no current plans to do so since the Ulm University of Applied Sciences (HSU) is the only contributor.

The suitability for other projects on this Tier must be considered through the runtime of the project. New project candidates may come up during the second open call thanks to its focus on applications (i.e. Tier 3) and therefore a need for RobMoSys to come up with concrete software building blocks on Tier 3.

### 4.4.4 Cross-Cutting Eclipse projects

RobMoSys aims at the realization of a virtual integration platform built upon existing tools and standards for the development of robotic systems. Concretely, this means that the RobMoSys approach and structures can enable model exchange and collaborative development between different tools that offer complementary features to cover all facets of developing robotic systems. There are several kinds of cross-cutting activities that are linked through the RobMoSys composition structures, i.e. model-exchange is supported through the composition structures.

As an example, consider the integration needs of, e.g., safety engineers and system integrators who use different RobMoSys-compliant software, each one covering different portion of the design cycle. On one side, SmartSoft and its large set of software components can be used to define the system's functional architecture. Then, a representative view of the system can be exported from SmartSoft and imported in Papyrus4Robotics, where a safety module can be used to perform dysfunctional analysis on the architecture's key components, including Hazard Analysis and Risk Assessment (HARA), Failure Mode and Effects Analysis (FMEA) and Fault Tree Analysis (FTA).

The following activities connected to RobMoSys are suitable as individual Eclipse projects that then describe their level of conformance to other projects within the RobMoSys Eclipse project structure (see Fig. 5):

**Papyrus4Robotics**

Papyrus (https://www.eclipse.org/papyrus/) is an Eclipse-based, open-source modeling environment that supports model-based design according to standards such as OMG's UML, SysML, MARTE and many more. Papyrus has been used successfully in industrial projects and is the base platform for several industrial modeling tools—more information about Papyrus' use case stories is available here: https://www.eclipse.org/papyrus/testimonials.html. Papyrus implements the ISO/IEC 42010 standard for systems and software architecture descriptions and naturally supports domain-specific customization of architecture viewpoints, diagram notations and styles, properties views, etc.

Papyrus4Robotics is a framework that features a customized version of Papyrus, enriched with domain-specific languages (DSLs), diagrams, viewpoints and add-on tools dedicated to design and analysis of robotic systems. Papyrus4Robotics targets full conformance to RobMoSys's foundational principles of separation of roles and concerns. The framework enables robot-systems design based on

- modeling: it comes with its own realization of RobMoSys composition structures, based on UML, MARTE and RobotML
- analysis: it features dedicated modules/plugins for model-based assessment of system-level properties at early design phases;
- synthesis: it supports code generation from models of software architectures to several platform and operating systems, including embedded and real-time and DDS-based distributed systems as potential targets;
- model-exchange: it supports model-exchange to realize virtual integration and leverage existing RobMoSys baselines that offer additional design capabilities.

Papyrus is an Eclipse project (https://www.eclipse.org/papyrus/). Papyrus4Robotics is under development and will be released as Eclipse project.

See also: https://robmosys.eu/wiki/baseline:environment_tools:papyrus4robotics

**The SmartSoft World**

SmartSoft is an umbrella term for structures, tools and building blocks to build robotics systems: a systematic development methodology, best practices, implementations and software components. The SmartSoft world consists of the following parts (all available as open-source):

- Structures
  - The SmartSoft World is conform to the RobMoSys composition structures at Tier 1. SmartSoft comes with its RobMoSys-conformat implementation of meta-models. These structures are shipped with the SmartMDSD Toolchain (see below).
- Infrastructure
  - The SmartSoft World includes several open-source exchangeable reference implementations of the SmartSoft Framework for several platforms and operating systems (e.g. based on CORBA, ACE, DDS, OPC-UA).
  - They are maintained by Ulm University of Applied Sciences
  - and are conform to the RobMoSys composition structures.
- Tooling
  - The SmartMDSD Toolchain is an Integrated Development Environment (IDE) for robotics software development using model-driven software development. The underlying meta-models are conform to the RobMoSys structures. The SmartMDSD Toolchain therefore provides model-driven tooling that supports the RobMoSys approach. It supports the Tier 2 and the Tier 3 users in applying RobMoSys. It ensures that their contributions stay conform to RobMoSys and can be composed by others. The SmartMDSD Toolchain has been an open source project for the last six years and reached an adequate maturity level for productive use (TRL 6, see (Stampfer 2017; Stampfer et al. 2016) for a survey)
- Software Components
  - A collection of building blocks for immediate composition to new robotic systems is already available as open source components. See “Eclipse projects for RobMoSys Tier 3”.

An **Eclipse project is currently under preparation to include model-driven tooling for the SmartSoft World** with a particular focus on the SmartMDSD Toolchain. The SmartMDSD Toolchain already builds on Eclipse technology. The project is to support modeling of domain structures (support of Tier 2 users) and the creation and use of building blocks on Tier 3.

See also:

- http://www.servicerobotik-ulm.de
- https://robmosys.eu/wiki/baseline:environment_tools:smartsoft:start

# 5 RobMoSys as an Eclipse Working Group

The overall scope of the RobMoSys project to establish structures for a European software ecosystem for robotics suggests to consider establishing an Eclipse Working Group. The granularity of such a working group may be the same as described in the mapping of the RobMoSys ecosystem to Eclipse projects. There is a special need for a “vendor-neutral governance structure” (see section “Eclipse Working Groups”).

RobMoSys is a suitable candidate to consider for establishing an Eclipse Working Group as the different Tier of the RobMoSys ecosystem would benefit from building this specific community, and potentially adapting the Eclipse governance.

# 6 RobMoSys Dissemination Activities in the Eclipse community

Several dissemination activities were undertaken in order to connect RobMoSys with the Eclipse community, to discuss potential Eclipse projects, and to discuss the establishment of ecosystems.

- Ansgar Radermacher. “RobMoSys”, Video Interview by Wayne Beaton, Eclipse Con France, Toulouse, 2017, https://www.youtube.com/watch?v=euW4yGTtmYg
- Dennis Stampfer, Alex Lotz, Matthias Lutz, Christian Schlegel. "RobMoSys and SmartSoft: Structures, tools and building blocks for robotics software development", Unconference Agora Meeting at EclipseCon Europe, Ludwigsburg, 2017.
- RobMoSys being present at the Eclipse Research booth at EclipseCon Europe 2017, Ludwigsburg, 2017.
- Presentation of the Eclipse ecosystem and how it is planned to be deployed for RobMoSys at ROS-Industrial conference 2017, Stuttgart 2017

# 7 Conclusion

The deliverable presents the Eclipse open source ecosystem and aligns the organization of the RobMoSys Ecosystem thereto. The deliverable argues why it is adequate to map the RobMoSys ecosystem into several Eclipse projects with a specific focus instead of having one all-in-one Eclipse project to contain the RobMoSys results. A generic mapping of RobMoSys into multiple focused projects is presented. Concrete Eclipse project candidates have been identified. The first Eclipse proposal is in preparation.

This document will be updated on an annual basis to further develop and report on the creation of Eclipse projects. The existing RobMoSys activities will continuously be evaluated for their suitability as Eclipse projects. Additionally, the results of the open call projects must be considered. These new contributions may either be included in already foreseen Eclipse projects or lead to new projects within the structure that was presented in this document.

# 8 References

RobMoSys Consortium. 2017a. “RobMoSys Deliverable D7.2: Business Models for The Ecosystem.” http://www.robmosys.eu.

———. 2017b. “RobMoSys Deliverable D7.3: Sustainability Plan.” https://doi.org/http://www.robmosys.eu.

Stampfer, Dennis. 2017. “Contributions to System Composition Using a System Design Process Driven by Service Definitions for Service Robotics (unpublished Work).”

Stampfer, Dennis, Alex Lotz, Matthias Lutz, and Christian Schlegel. 2016. “The SmartMDSD Toolchain: An Integrated MDSD Workflow and Integrated Development Environment (IDE) for Robotics Software.” *Journal of Software Engineering for Robotics (JOSER)* 7 (1):3–19. http://www.joser.org.