Beagle

Implementation Plan

Annika Berger, Joshua Gleitze, Roman Langrehr,
Christoph Michelbach, Ansgar Spiegler, Michael Vogt

10th of January 2016

at the Department of Informatics
Institute for Program Structures and Data Organization (IPD)

Reviewer:	Jun.-Prof. Dr.-Ing. Anne Koziolek
Advisor:	M.Sc. Axel Busch
Second advisor:	M.Sc. Michael Langhammer

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

Implementation Plan

Each task will be assigned to one person and another person will write tests for it. Except for the prototypes: they do not need automated tests.

Preparation (2015-12-24 – 2016-01-10)

For the parts of our project depending on the Eclipse API we decided to create prototypes to learn how to use the API. As knowledge about used APIs is essential for a software's design, we decided to already create prototypes in the design phase.

Task:	Prototype for Context Menus
Nr:	#010
Description/Classes:	Write a simple Eclipse plugin that adds all context menus specified in the SRS GUI model to Eclipse by using Eclipse extension points. Show a sample dialog when the user clicks on a context menu.
Depends on:	–

Task:	Prototype for Eclipse Extension Points
Nr:	#020
Description/Classes:	Write a simple Eclipse plugin that offers an extension point for measurement tools and allows other plugins to offer a measurement tool. The plugin should display the available measurement tools. Create a plugin with a measuremnt tool stub that uses the extension point.
Depends on:	–

Task:	Prototype for GUI
Nr:	#030
Description/Classes:	Create an Eclipse plugin that shows a wizard as described in the SRS GUI model. The wizard should show some demonstration content.
Depends on:	–

Week 1 (2016-01-11 – 2016-01-17)

Task:	SEFF classes
Nr:	#040
Description/Classes:	Implement `SEFFLoop`, `SEFFBranch`, `ResourceDemandingInternalAction`.
Depends on:	–

Task:	PCM Repository loader
Nr:	#050
Description/Classes:	Create a class that provides all information from the PCM which are relevant for Beagle and a factory that uses these information to create the `SEFFLoops`, `SEFFBranches` and `ResourceDemandingInternalActions`.
Depends on:	#040 (SEFF classes)

Task:	Blackboard
Nr:	#060
Description/Classes:	Implement `Blackboard`.
Depends on:	– (Not depending on #050, as the `Blackboard` only needs the class stubs for `SEFFLoop`, `SEFFBranch`, `ResourceDemandingInternalAction` and the `ParameterisationDependentMeasurementResult` subclasses and not their functionality).

Task:	Blackboard Views
Nr:	#070
Description/Classes:	Implement `Read-OnlyBlackboardView`, `AnalyserBlackboardView`, `MeasurementControllerBlackboardView`, `FinalJudgeBlackboardView`.
Depends on:	– (Not depending on #060, as the Blackboard Views only need the blackboards method stubs.)

Task:	Integrate prototype for Context Menus
Nr:	#080
Description/Classes:	Integrate the prototype for Context Menus into Beagle.
Depends on:	#010

Task:	Integrate prototype for Eclipse Extension Points
Nr:	#090
Description/Classes:	Integrate the prototype for Eclipse Extension Points into Beagle.
Depends on:	#020

Task:	Prototype for GUI
Nr:	#100
Description/Classes:	Integrate the prototype for Context Menus into Beagle.
Depends on:	#030

Week 2 (2016-01-18 – 2016-01-24)

Task:	Implement Evaluable Expressions
Nr:	#110
Description/Classes:	Implement all subclasses of the interface `EvaluableExpression`. (Package “Evaluable Expressions”)
Depends on:	–

Task:	Implement Measurment Results
Nr:	#120
Description/Classes:	Implement `ParameterisationDependentMeasurementResult` and all of its subclasses. (Package “Measurement”)
Depends on:	–

Task:	Implement Blackboard Controller
Nr:	#130
Description/Classes:	Implement `BeagleController` and `MeasurementController`.
Depends on:	#060 (`Blackboard`) and #070 (Blackboard Views)

Week 3 (2016-01-25 – 2016-01-31)

Task:	Kieker measurement tool
Nr:	#140
Description/Classes:	Build a `MeasurementTool` executing Kieker.
Depends on:	#120 (Measurment Results), #060 (`Blackboard`), #070 (Blackboard Views) and #040 (SEFF classes)

Task:	Averaging Final Judge
Nr:	#150
Description/Classes:	Build a `FinalJudge`, which takes as final `EvaluableExpression` for all `MeasurableSeffElements` the average of all available measurement results.
Depends on:	–

Optional: Build more, better `MeasurementTools`, `ResultAnalysers` and `FinalJudges`.

Week 4 (2016-02-01 – 2016-02-07)

Reserved for tasks that take longer than expected.

Optional: Build more, better `MeasurementTools`, `ResultAnalysers` and `FinalJudges`.