ICT PROJECT 258109
Monitoring Control for Remote Software Maintenance

Project number: 258109
Project Acronym: FastFix
Project Title: Monitoring Control for Remote Software Maintenance
Call identifier: FP7-ICT-2009-5
Instrument: STREP
Thematic Priority: Internet of Services, Software, Virtualisation
Start date of the project: June 1, 2010
Duration: 33 months

D2.7: Overview of FastFix Deliverables

Lead contractor: TUM
Author(s): Dennis Pagano, Luis Ballester
Submission date: February 2013
Dissemination level: Public

Project co-founded by the European Commission under the Seventh Framework Programme
© S2, TUM, LERO, INESC ID, TXT, PRODEVELOP
Abstract: This document provides an overview of the FastFix deliverables. Its objective is to briefly summarize the FastFix approaches and their advantages, and to help understanding which information to find in which FastFix deliverable, illustrating how the different deliverables are interconnected.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Introduction and Organization of the Document</td>
<td>6</td>
</tr>
<tr>
<td>0.1</td>
<td>FastFix Project Structure</td>
<td>6</td>
</tr>
<tr>
<td>0.2</td>
<td>Work Package Objectives</td>
<td>7</td>
</tr>
<tr>
<td>0.3</td>
<td>FastFix Approaches and Advantages</td>
<td>9</td>
</tr>
<tr>
<td>0.3.1</td>
<td>Context Observation</td>
<td>9</td>
</tr>
<tr>
<td>0.3.2</td>
<td>Event Correlation and Pattern Mining</td>
<td>9</td>
</tr>
<tr>
<td>0.3.3</td>
<td>FaultReplication</td>
<td>9</td>
</tr>
<tr>
<td>0.3.4</td>
<td>Patch Generation and Self-Heling</td>
<td>9</td>
</tr>
<tr>
<td>0.4</td>
<td>Document Outline</td>
<td>10</td>
</tr>
<tr>
<td>1</td>
<td>Phase 1: State of the Art Survey</td>
<td>11</td>
</tr>
<tr>
<td>1.1</td>
<td>D2.1: State-of-the-art in monitoring control for remote maintenance</td>
<td>11</td>
</tr>
<tr>
<td>1.2</td>
<td>D3.1: State-of-the-art Context Elicitation, Context Modeling and User</td>
<td>11</td>
</tr>
<tr>
<td>1.3</td>
<td>D4.1: State-of-the-art of event correlation and event processing</td>
<td>12</td>
</tr>
<tr>
<td>1.4</td>
<td>D5.1: State of the art in fault replication and test automation</td>
<td>12</td>
</tr>
<tr>
<td>1.5</td>
<td>D6.1: State-of-the-art in Self-Healing and Patch Generation</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>Phase 2: Definition of Conceptual Models</td>
<td>13</td>
</tr>
<tr>
<td>2.1</td>
<td>D3.2: Conceptual Model for Context Observation and User Profiling</td>
<td>13</td>
</tr>
<tr>
<td>2.2</td>
<td>D4.3: Conceptual models of the event processor and pattern recognition</td>
<td>13</td>
</tr>
<tr>
<td>2.3</td>
<td>D5.2: Conceptual Model of Fault Replication Platform</td>
<td>14</td>
</tr>
<tr>
<td>2.4</td>
<td>D6.2: FastFix Conceptual Models for Self-Healing and Patch Generation</td>
<td>14</td>
</tr>
<tr>
<td>3</td>
<td>Phase 3: Architecture, Requirements, Specifications</td>
<td>15</td>
</tr>
<tr>
<td>3.1</td>
<td>D2.2: Integration Plan and Technical Project Guidelines</td>
<td>15</td>
</tr>
<tr>
<td>3.2</td>
<td>D2.3: User Requirements and Conceptual Architecture</td>
<td>15</td>
</tr>
<tr>
<td>3.3</td>
<td>D2.4: Architecture Changes and Change Rationales</td>
<td>16</td>
</tr>
<tr>
<td>3.4</td>
<td>D4.2: Domain Specific Language for Event Correlation</td>
<td>16</td>
</tr>
<tr>
<td>3.5</td>
<td>D7.1: Global performance, security and privacy requirements</td>
<td>16</td>
</tr>
<tr>
<td>3.6</td>
<td>D7.2: Global threat model analysis document</td>
<td>17</td>
</tr>
<tr>
<td>4</td>
<td>Phase 4: Prototype Construction and Validation</td>
<td>18</td>
</tr>
<tr>
<td>4.1</td>
<td>D3.3: 1st Prototype of the Context Observer</td>
<td>18</td>
</tr>
<tr>
<td>4.2</td>
<td>D3.4: 1st Prototype of the User Profiler</td>
<td>19</td>
</tr>
<tr>
<td>4.3</td>
<td>D3.5: 1st Prototype of the Error Reporting</td>
<td>19</td>
</tr>
<tr>
<td>4.4</td>
<td>D3.6: Refined and Integrated Version of Context Observer, User Profiler</td>
<td>19</td>
</tr>
<tr>
<td></td>
<td>and Error Reporting</td>
<td>19</td>
</tr>
<tr>
<td>4.5</td>
<td>D4.4: 1st iteration prototype of the Event Processor</td>
<td>19</td>
</tr>
<tr>
<td>4.6</td>
<td>D4.5: 1st iteration prototype of the pattern mining module</td>
<td>19</td>
</tr>
</tbody>
</table>
D2.7: Overview of FastFix Deliverables

4.7  D4.6: Second refined prototype of the event correlation component
4.8  D5.3: 1st Prototype of the Execution Recorder/Replayer Tool
4.9  D5.4: 2nd Prototype of the Execution Recorder/Replayer Tool
4.10 D6.3: 1st Prototype of the Self-Healing and Patch Generation Component
4.11 D6.4: 2nd Prototype of the Self-Healing and Patch Generation Component
4.12 D6.5: 3rd Prototype of the Self-Healing and Patch Generation Component
4.13 D6.6: 4th Prototype of the Self-Healing and Patch Generation Component

5  Phase 5: Prototype Integration and Final Platform

5.1  D2.5: 1st Integrated Version of FastFix Platform
5.2  D2.6: Final Version of FastFix Platform

6  Phase 6: Trials and Validation

6.1  D7.5: Security Audit Results
6.2  D8.1: Trial and Validation Plan
6.3  D8.2: As is Analysis of Software Maintenance
6.4  D8.3: FastFix Formative Evaluation
6.5  D8.4: Summative Evaluation Report

7  Continuous: Project Management and Quality Assurance

7.1  D1.1: FastFix Management Manual
7.2  D1.2: 1st Periodic Management and Quality Report
7.3  D1.3: 2nd Periodic Management and Quality Report
7.4  D1.4: 3rd Periodic Management and Quality Report
7.5  D1.5: FastFix Final Report

8  Continuous: Dissemination and Exploitation

8.1  D9.1: FastFix Brochure and Public Website
8.2  D9.2: Dissemination Plan and Reports
8.3  D9.3: Exploitation Plan and Reports
8.4  D9.4: Collaboration Plan

Page 4 of 28
List of Figures

0.1 FastFix work packages .............................................................. 6
0.2 FastFix deliverable phases ............................................................ 8
1.1 Deliverables in FastFix Phase 1 ...................................................... 11
2.1 Deliverables in FastFix Phase 2 ...................................................... 13
3.1 Deliverables in FastFix Phase 3 ...................................................... 15
4.1 Deliverables in FastFix Phase 4 ...................................................... 18
5.1 Deliverables in FastFix Phase 5 ...................................................... 22
6.1 Deliverables in FastFix Phase 6 ...................................................... 23
7.1 Deliverables of FastFix Project Management and Quality Assurance Activities ...................................................... 25
8.1 Deliverables of FastFix Dissemination and Exploitation Activities ...................................................... 27
0 Introduction and Organization of the Document

This section introduces the FastFix project structure, detailing on the work package objectives, and illustrates the organization of the document.

0.1 FastFix Project Structure

The FastFix project is organized in nine different work packages (WP), illustrated in Figure 0.1, which belong to three main areas: management (WP1), research and technological development (WP2-8), as well as dissemination and exploitation of the obtained results (WP9). Deliverables in the management area mainly report on the project situation, while deliverables in the research area typically include conceptual model descriptions.
D2.7: Overview of FastFix Deliverables

and refinements, as well as iterative descriptions of the implemented prototypes and their interaction with other FastFix components.

0.2 Work Package Objectives

WP1 is concerned with project management and assures the quality of the FastFix results. The deliverables of this work package regard non-public project internals, and thus are not described in this document. Work packages 2 through 8 deliver the actual research and technological contributions to the project. Among them, WP2 is responsible for the conceptual framework of FastFix, its architecture, and the integration of the developed components. WP3-6 represent the main research lines of FastFix: context observation and user modeling, event correlation, fault replication, as well as patch generation and self healing. WP7 is concerned with non-functional requirements for the platform, specifically performance, security, and privacy. Last, WP8 is responsible for planning and executing trials which validate the developed concepts and components. WP9 is concerned with dissemination activities and the exploitation of the FastFix results. Its objectives are to produce and disseminate promotional material for FastFix, to effectively disseminate FastFix achievements, to assess the market for FastFix results, to coordinate FastFix with related initiatives, and to effectively plan and organize exploitation of FastFix results.
D2.7: Overview of FastFix Deliverables

Figure 0.2: FastFix deliverable phases.
D2.7: Overview of FastFix Deliverables

0.3 FastFix Approaches and Advantages

0.3.1 Context Observation

The context observation component of FastFix monitors applications and their users. Detailed information about the dynamic execution of client-server applications and actions of users interacting with an application are collected. This information is the basis for other FastFix features, including event correlation, fault replication, and self-healing. User actions and application events are represented as context events in a semantic event taxonomy. This allows for uniform reasoning about events detected by an extensible set of sensors. FastFix sensors collect different types of events and target different application technologies, which demonstrates the possibility to instrument different application types and implementation technologies.

0.3.2 Event Correlation and Pattern Mining

FastFix uses event correlation techniques for software error detection and cause identification. To this end, it processes clouds of events generated in several application instances, both in different execution platforms and in different sessions in the same platform. FastFix uses pattern matching and semantic information about the events to identify error symptoms and causes. To this end, FastFix holds information about patterns of errors in ontologies, in particular the conditions to be satisfied by a set of events in order to represent a fault, and uses this information within abstract rules to detect these patterns. Patterns of errors can be maintained by maintenance experts without in-depth knowledge of FastFix, by using a graphical interface developed for this purpose. In addition, FastFix includes a pattern mining mechanism, which analyzes traces of runtime events using frequent pattern mining algorithms with the purpose of learning the normal behavior of the application. Then, it compares and contrasts streams of events against the expected behavior. As a result, FastFix is able to recommend suspect patterns to maintenance engineers, who may acknowledge them as relevant.

0.3.3 Fault Replication

FastFix provides advances in fault replication by providing automatic, high fidelity recording and replaying of applications. Software errors in client devices appear in the maintenance team’s ticketing system ready to be replayed. Furthermore, FastFix addresses the privacy concerns that prevent many users from submitting error reports. FastFix anonymizes error reports by exploring the structure of applications in order to minimize the leaked personal information. FastFix searches for alternative execution paths and also leverages the event-based structure of GUI applications in order to find the minimum possible error report that still replicates the clients’ fault.

0.3.4 Patch Generation and Self-Healing

The FastFix self-healing component implements a model-based self-healing approach for software applications. This component handles legacy code written in Java, and automatically generates models and instrumentation, equipping the application with monitoring...
and runtime decision facilities. Moreover when a runtime exception occurs, the application generates logs that the self-healing component analyse together with the application models in order to automatically generate patches. Patches represent new models that are used by the application at runtime for decision making. With these models, behaviors leading to the previously observed exception are now prevented. The main strength of the self-healing component is that it provides a complete solution that automates patch generation as well as equipping legacy application with self-healing properties. Although focused on automation, the component also allows for input from application experts, guiding patch generation to relevant solutions.

0.4 Document Outline

The following sections give an overview of the FastFix deliverables in each of the work packages. In this document we present the FastFix deliverables according to the different project phases. As illustrated in Figure 0.2, these include six consecutive phases for the technical contributions, as well as two continuous phases for project management, dissemination, and exploitation.

In the first phase of the project (Section 1), we surveyed the state of the art in the technical research lines. In the second phase of the project (Section 2), we developed and refined conceptual models for the four technical research lines. In the third phase of the project (Section 3), we specified requirements, designed the FastFix platform, and established technical project guidelines and integration means. The fourth phase of the project (Section 4) focused on the implementation of functional prototypes and the validation of the conceptual feasibility. The fifth phase of the project (Section 5) focused on the integration of the functional prototypes in the FastFix platform as a whole. The prototypes, as well as the resulting, integrated FastFix platform can be accessed in the project repository on SourceForge. The sixth phase of the project (Section 6) focused on evaluating the FastFix platform using industrial trials with real commercial applications and evaluation studies. Section 7 covers the deliverables produced during the continuous activities of project management and quality assurance. Finally, Section 8 summarizes the documents delivered during the continuous dissemination and exploitation activities.

1https://svn.code.sf.net/p/fastfixrsm/code/trunk
1 Phase 1: State of the Art Survey

<table>
<thead>
<tr>
<th>Phase 1: State of the art survey</th>
</tr>
</thead>
<tbody>
<tr>
<td>• D2.1: State-of-the-art in monitoring control for remote maintenance</td>
</tr>
<tr>
<td>• D3.1: State-of-the-art Context Elicitation, Context Modeling and User Modeling</td>
</tr>
<tr>
<td>• D4.1: State-of-the-art of event correlation and event processing</td>
</tr>
<tr>
<td>• D5.1: State of the art in fault replication and test automation</td>
</tr>
<tr>
<td>• D6.1: State-of-the-art in Self-Healing and Patch Generation</td>
</tr>
</tbody>
</table>

The first phase of the project (illustrated in Figure 1.1) was concerned with exploring and analyzing the state of the art and state of the practice of the different research lines included in FastFix. Specific attention was payed to comparing the state of the art with the objectives of the FastFix project.

1.1 D2.1: State-of-the-art in monitoring control for remote maintenance

This report summarizes the state of research and practice in the area of software maintenance with a particular emphasis on remote software maintenance. It summarizes models and processes used in classical software maintenance and remote software maintenance. Further, we describe approaches and tools in software-fault monitoring. Concluding, we identify research islands within remote software maintenance and propose how these islands should be merged beneficially.

1.2 D3.1: State-of-the-art Context Elicitation, Context Modeling and User Modeling

This report describes the state of the art of context awareness and user modeling, including the elicitation, processing and modeling of context. Specific attention is given to main issues in context awareness such as privacy, incompleteness, and ambiguity. We discuss how context information relevant for software maintenance can be collected. We also
D2.7: Overview of FastFix Deliverables

discuss how context in general and users in particular can be modeled based on semantic
technologies and machine learning and compare both approaches. Finally, we present first
uses of context in the domain of remote software maintenance.

1.3 D4.1: State-of-the-art of event correlation and
event processing

This document examines the state of research and practice regarding event correlation
and its application to software maintenance with a focus on the FastFix requirements.
We describe current event correlation techniques as well as its use in the field of software
error detection and cause identification.

1.4 D5.1: State of the art in fault replication and test
automation

This document examines the state of the art of fault replication and test automation and
relates them to the goals of the FastFix project. It presents central concepts regarding
the foundations of state replication as well as the main ideas in test automation. Further,
the most relevant academic and commercial fault replication and test automation system
are presented and their relation to FastFix is explored.

1.5 D6.1: State-of-the-art in Self-Healing and Patch
Generation

This document introduces current self-healing approaches as well as related researches
such as fault-tolerance, automatic diagnosis and automatic repair. The latter works relate
to automatic patch generation.
2 Phase 2: Definition of Conceptual Models

In the second phase of the FastFix project (illustrated in Figure 2.1), we developed and refined conceptual models for the four technical research lines, WP3-6, which provide the basis for implementing the corresponding components.

2.1 D3.2: Conceptual Model for Context Observation and User Profiling

This document describes a high level design for the context observation and user profiling components of FastFix. We discuss the main requirements for context observation and user profiling, and present an approach that allows application independent context observation, based on context events and interpreters, which process these events in order to generate knowledge useful for user profiling, event correlation, and fault replication. We also present a model for determining the relevance of the monitored information when creating user profiles.

The document extension discusses the relation between FastFix and TeamWeaver\(^1\) and details motivation behind and relevance of the user profiler component.

2.2 D4.3: Conceptual models of the event processor and pattern recognition

In this document we give a detailed description of the conceptual model and technologies needed to implement the event correlation module requirements.

\(^1\)http://www.teamweaver.org
In the first part, an ontology of failures is described, based on the output of other deliverables (D2.3, D3.2, D4.1, D4.2, and D6.2) and on consortium experience. Based on this structure, we define a mechanism to represent cause-effect relationships in a way that allows an easy adaptation to information sent by the context observer and user profiler (WP3). We further describe how to employ pattern recognition over the stream of events, in order to identify symptoms of failure before they occur and to detect performance degradation trends, with the goal to achieve preventive maintenance.

In the second part of the document we present a more detailed overview of the technologies involved in the conceptual model. We illustrate the evolution of the prototype based on the results of a summative evaluation in terms of performance. We also show an overview of the third party libraries needed to implement the event correlation requirements.

2.3 D5.2: Conceptual Model of Fault Replication Platform

This document describes the concepts and architecture, which underlie the fault replication functionality in FastFix. Starting with a set of design goals, the fault replication system is detailed in its steps (pre-deployment application instrumentation, execution monitoring, error report creation and submission, and execution replay). For each of these steps the main goals and hurdles are discussed and specific solution designs are presented. Along the way, all features are also placed in relation to the whole FastFix architecture.

2.4 D6.2: FastFix Conceptual Models for Self-Healing and Patch Generation

This document describes the conceptual models for self-healing and patch generation within the FastFix project and presents challenges and solutions related to implementing these models. In this conceptual model, the FastFix platform automatically equips the target application with autonomic features. This includes self-monitoring capabilities as well as self-awareness through the embedding of models. When an issue occurs, the state of the application is logged. These logs together with the models of the application are used in order to automatically generate new models of the application where previously undesired behaviors are avoided. These new models are embedded in the application at runtime where decisions can be made to ensure correct system behaviors.
3 Phase 3: Architecture, Requirements, Specifications

In the third phase of the FastFix project (illustrated in Figure 3.1), we concentrated on specifying requirements, designing the FastFix platform, and established technical project guidelines and integration means.

3.1 D2.2: Integration Plan and Technical Project Guidelines

This deliverable describes the project standards of FastFix. It defines the development and integration processes as well as quality assurance activities. We describe a continuous integration approach that reduces integration risks and removes dependencies between releases and integration activities. This report also describes the project infrastructure and tools used in the project. Additionally it introduces standards and guidelines to facilitate tool usage and standardize developed artifacts in FastFix.

3.2 D2.3: User Requirements and Conceptual Architecture

In this document we give an overview about the FastFix approach. We describe scenarios where the maintenance of a software application can be supported or automated by monitoring its execution and context, and present a concrete technical blueprint of FastFix.
We present requirements for the FastFix system derived from interviewing and observing maintenance engineers and analyzing the desired functionality, which covers three aspects: (1) the classification of error types, (2) the FastFix error handling strategies, i.e. the different approaches that handle the specific error types, and (3) scenarios from real world applications of the consortium companies, combining error types, types of target applications, and underlying technologies. Further, we illustrate the high-level architecture of the FastFix system and introduce the main ontologies, which formalize the conceptual model and provide the data representation mechanism in FastFix. In addition, we investigate how artificial intelligence techniques can be applied and combined within the FastFix system in order to meet the requirements.

In the extension of the document we illustrate the consortium’s effort to refine the scenarios which guide the development of the FastFix system and represent demonstrators for the main use cases. Specifically, we illustrate our efforts to (a) understand the error landscape in research and practice, and (b) rescope and refine the project scenarios. We report on a literature review and internal FastFix case studies. We use these results to establish a reasonable project focus, describing six project scenarios for relevant error types. We illustrate these scenarios by concrete real errors in industrial partners’ existing applications. Further, we indicate which FastFix error handling strategies can be used to mitigate the errors, and show how the according functionality is realized in detail.

3.3 D2.4: Architecture Changes and Change Rationales

In this document, we substantiate the overall conceptual FastFix architecture, which was described in D2.3. We describe changes we have made, and illustrate our rationales for these changes. Thus, we provide an updated overview of the system’s main components as well as a description of their most important public interfaces.

3.4 D4.2: Domain Specific Language for Event Correlation

This document summarizes steps to create a domain specific language (DSL) and defines a DSL for software execution monitoring rules with the goal of making it understandable for both application developers and machines and generic enough to be independent of application, system, and runtime environment.

3.5 D.7.1: Global performance, security and privacy requirements

This document examines the state of research and practice regarding performance optimization on run-time monitoring, paying special attention to reduce the footprint on monitored applications and the possibility to rely on resources provided by multicore processors. Moreover, it explores security requirements, in particular investigating confi-
dentilation solutions for network traffic, confidentiality solutions for long-term storage of key data or privacy concerns, specially for personal data.

3.6 D7.2: Global threat model analysis document

This document describes identified threats, establishing a taxonomy of threats which is compared to real data obtained in similar environments.
4 Phase 4: Prototype Construction and Validation

<table>
<thead>
<tr>
<th>Phase 4: Prototype construction and validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>• D3.3: 1st Prototype of the Context Observer</td>
</tr>
<tr>
<td>• D3.4: 1st Prototype of the User Profiler</td>
</tr>
<tr>
<td>• D3.5: 1st Prototype of the Error Reporting</td>
</tr>
<tr>
<td>• D3.6: Refined and Integrated Version of Context Observer, User Profiler, and Error Reporting</td>
</tr>
<tr>
<td>• D4.4: 1st iteration prototype of the Event Processor</td>
</tr>
<tr>
<td>• D4.5: 1st iteration prototype of the pattern mining module</td>
</tr>
<tr>
<td>• D4.6: Second refined prototype of the event correlation component</td>
</tr>
<tr>
<td>• D5.3: 1st Prototype of the Execution Recorder / Replayer Tool</td>
</tr>
<tr>
<td>• D5.4: 2nd Prototype of the Execution Recorder / Replayer Tool</td>
</tr>
<tr>
<td>• D6.3: 1st Prototype of the Self-Healing and Patch Generation Component</td>
</tr>
<tr>
<td>• D6.4: 2nd Prototype of the Self-Healing and Patch Generation Component</td>
</tr>
<tr>
<td>• D6.5: 3rd Prototype of the Self-Healing and Patch Generation Component</td>
</tr>
<tr>
<td>• D6.6: 4th Prototype of the Self-Healing and Patch Generation Component</td>
</tr>
</tbody>
</table>

Figure 4.1: Deliverables in FastFix Phase 4.

The fourth phase of the FastFix project (illustrated in Figure 4.1) focused on the implementation of functional prototypes and the validation of the conceptual feasibility. These prototypes can be accessed in the project repository.

4.1 D3.3: 1st Prototype of the Context Observer

This document summarizes the development work performed for the context observer. It presents, by means of a tutorial, how the monitoring component of the context system can be used to create sensors which provide context information to the other FastFix components. The source code can be accessed from the FastFix repository.

1https://svn.code.sf.net/p/fastfixrsm/code/trunk
4.2 D3.4: 1st Prototype of the User Profiler

This document summarizes how the user profiler component of FastFix works, how it is implemented and how to execute it. It is a supplement to the source code of the first version of the user profiler which can be accessed from the FastFix repository.

4.3 D3.5: 1st Prototype of the Error Reporting

This document describes the design and usage of the communication and error reporting components of the FastFix project’s platform. These components are used for client-server messaging and for sending error reports and auxiliary files to the maintenance team in a FastFix installation. This document is a supplement to the source code of the first version of the user profiler which can be accessed from the FastFix repository.

4.4 D3.6: Refined and Integrated Version of Context Observer, User Profiler, and Error Reporting

This document reports on the context observer, error reporting, and user profiler components of FastFix. We give an overview how each component works, describe how they interact to realize FastFix functionality, and summarize the changes since the first prototypes. This document is a supplement to the source code which can be accessed from the FastFix repository.

4.5 D4.4: 1st iteration prototype of the Event Processor

This document reports on the implementation of the 1st prototype of the event correlation component, its architecture, and its main components. This document is a supplement to the source code which can be accessed from the FastFix repository.

4.6 D4.5: 1st iteration prototype of the pattern mining module

This document reports on the implementation of the 1st prototype of the pattern mining module, a subcomponent specialized in the identification of event patterns that allow for failure prevention and performance degradation. The idea of the pattern mining feature is centered on the design of a mechanism to acquire new software maintenance knowledge about existing errors, from the events flowing in the system. This module is strongly connected to the event correlation system (D4.4), since the kind of knowledge that the system aims to acquire will feed the event correlation system back, taking advantage of the new mined patterns to detect new errors the FastFix system was not able to detect previously with the existing knowledge. This document is a supplement to the source code which can be accessed from the FastFix repository.
4.7 D4.6: Second refined prototype of the event correlation component

This document adds on previous deliverables (D4.2, D4.3, and D4.4), where both event processor component and pattern mining module have been presented. It aims to describe the integration of both components into a single prototype of the event correlation system, showing the benefits of its interaction. The pattern mining module provides the feature of automatic error pattern discovery, which boils down to the delivery of possible error patterns that might be useful to the FastFix user. Once validated, the event correlation system takes the lead, providing a mechanism to automatically detect any of these error patterns. Additionally, the document describes the design and usage of the pattern management interface, which is the user interface to manage mined and user-created patterns, one of the available FastFix tools to support the software maintenance team to detect in “real time” the conditions that lead to an error or unexpected behavior, collecting the relevant data to help determine the cause of such situations.

4.8 D5.3: 1st Prototype of the Execution Recorder/Replayer Tool

This document describes the design and usage of the execution recorder/replayer components of FastFix. These components are used for providing enhanced error reporting capabilities by recording application execution and therefore providing a reliable fault replication mechanism to help maintenance teams using the FastFix platform.

4.9 D5.4: 2nd Prototype of the Execution Recorder/Replayer Tool

This document describes the design and usage of the 2nd iteration of the execution recorder/replayer components of FastFix, called fault replication. It presents the improvements since the version described in deliverable D5.3. The described components provide an enhanced, automated and anonymized error reporting mechanism by recording application execution and automatic generation of error reports. This mechanism allows developers to reliably reproduce field failures in their own environments. This document is a supplement to the source code which can be accessed from the FastFix repository.

4.10 D6.3: 1st Prototype of the Self-Healing and Patch Generation Component

This document presents a first implementation of the self-healing and patch generation component. It introduces a bootstrapped implementation of the different phases involved in the approach: model extraction, instrumentation, patch generation and runtime supervision. It also details the challenges that such an implementation must face in term of scalability and relevance.
4.11 D6.4: 2nd Prototype of the Self-Healing and Patch Generation Component

This document presents improvements brought to the FastFix self-healing component since the version described in deliverable D6.3. These improvements are manifold and regard all the different aspects of the component: model extraction, supervision mechanism, runtime supervision as well as patch generation. The main improvement compared to the version presented in D6.3 resides in a generic way to select the part of the system that will be modeled and instrumented. This approach makes it possible to not only tackle java application using the SWING interface and also eclipse based application such as Moskitt, which is developed by Prodevelop and is used as a case study in the project.

4.12 D6.5: 3rd Prototype of the Self-Healing and Patch Generation Component

This document presents improvements brought to the FastFix self-healing component since the version described in deliverable D6.3 and D6.4. Model extraction, patch generation and runtime supervision will be extended in order to take into account application variables and user inputs. This addition will increase the accuracy of the patch generation and runtime supervision mechanisms.

4.13 D6.6: 4th Prototype of the Self-Healing and Patch Generation Component

This document presents improvements of the FastFix self-healing component. More specifically this deliverable provides a different patching strategy, complementary to the one considered in previous iterations. This iteration of the prototype also makes it possible to introduce expertise in the loop through configuration files. Although the encoding of this expertise is not an automated process, the overhead induced by it is kept to minimum. Moreover such expertise leads self-healing and patch generation towards relevant solutions.
5 Phase 5: Prototype Integration and Final Platform

The fifth phase of the project (illustrated in Figure 5.1) focused on the integration of the functional prototypes in the FastFix platform as a whole. The integrated FastFix platform can be accessed in the project repository.

5.1 D2.5: 1st Integrated Version of FastFix Platform

This document describes the first integrated version of the FastFix platform. It is a supplement to the source code, which can be accessed on the FastFix repository. It gives a conceptual overview of the platform as a whole, and describes how the FastFix platform is typically deployed and used.

5.2 D2.6: Final Version of FastFix Platform

This document describes the final version of the FastFix platform, pinpointing the changes made after the first version described in D2.5. It is a supplement to the source code, which can be accessed on the FastFix repository. It gives a conceptual overview of the platform as a whole, and describes how the FastFix platform is typically deployed and used.

\[\text{Figure 5.1: Deliverables in FastFix Phase 5.}\]

---

1. https://svn.code.sf.net/p/fastfixrsm/code/trunk
6 Phase 6: Trials and Validation

<table>
<thead>
<tr>
<th>Phase 6: Trials and validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>• D7.5 Security audit results</td>
</tr>
<tr>
<td>• D8.1: Trial and Validation Plan</td>
</tr>
<tr>
<td>• D8.2: As is Analysis of Software Maintenance</td>
</tr>
<tr>
<td>• D8.3: FastFix Formative Evaluation</td>
</tr>
<tr>
<td>• D8.4: Summative evaluation report</td>
</tr>
</tbody>
</table>

Figure 6.1: Deliverables in FastFix Phase 6.

The sixth phase of the FastFix project (illustrated in Figure 6.1) focused on evaluating the FastFix platform. To this end, we set up industrial trials with real commercial applications and performed evaluation studies.

6.1 D7.5: Security Audit Results

The integrated FastFix platform must be evaluated, not only in terms of performance or scalability, but also in terms of security. For this purpose, a security audit strategy is designed, based on the requirements already defined in D7.1. This document aims to describe both the security audit plan and the obtained results. Finally, conclusions and recommendations are included, in order to be taken into account for the last refinements of the platform.

6.2 D8.1: Trial and Validation Plan

This document describes the trial and validation plan for the FastFix project. It outlines an evaluation strategy based on a formative phase during which metrics are derived from information obtained from the software development process of the FastFix platform, and a summative phase which addresses the validation from several angles, including experiments to derive behaviour metrics, user experiments and interviews to evaluate user satisfaction, and pilot application trials to try out FastFix in real world settings.

6.3 D8.2: As is Analysis of Software Maintenance

This document describes the current state of software maintenance at the industrial partners participating in the FastFix project. It covers the processes in place, the cost of operations for maintenance aspects, a study of the type of problems found in industrial
applications, products, applications and systems in use, a business analysis of the current state of the maintenance processes and a summary of benefits that FastFix can provide. The analysis of the state of the software maintenance allows to find strengths and opportunities that FastFix can exploit. This study also allows FastFix to better adapt to the current state of software maintenance in industrial organizations.

6.4 D8.3: FastFix Formative Evaluation

This document describes the formative evaluation that has taken place during the development of the FastFix project. This includes the definition of the process, the decisions taken and the rationale, and illustrates how the process has been applied to the development of software artefacts and the improvements that the process has brought to the development of the platform.

6.5 D8.4: Summative Evaluation Report

The summative report collects the progress of the project, the results achieved and provides a summary of the evaluation performed on the results. The evaluation includes, but is not limited to, the execution of several trials and studies to measure and analyse the overall benefit of FastFix in industrial software environments. The analysis includes performance impact of the platform on existing applications and systems, benefits seen in the detection of errors and problems, and improvements for the resolution of errors.
7 Continuous: Project Management and Quality Assurance

<table>
<thead>
<tr>
<th>Continuous: Project Management and Quality Assurance</th>
</tr>
</thead>
<tbody>
<tr>
<td>• D1.1: FastFix Management Manual</td>
</tr>
<tr>
<td>• D1.2: 1st Periodic Management and Quality Report</td>
</tr>
<tr>
<td>• D1.3: 2nd Periodic Management and Quality Report</td>
</tr>
<tr>
<td>• D1.4: 3rd Periodic Management and Quality Report</td>
</tr>
<tr>
<td>• D1.5: FastFix Final Report</td>
</tr>
</tbody>
</table>

Figure 7.1: Deliverables of FastFix Project Management and Quality Assurance Activities.

Project management activities and quality assurance of the FastFix results (illustrated in Figure 7.1) is the focus of the FastFix WP1. The deliverables of this work package regard non-public project internals.

7.1 D1.1: FastFix Management Manual

This document confirms the organization structure of the project and the procedures established to make it work in practice. It includes a description of the management structure, the project information system, and the quality control procedures.

7.2 D1.2: 1st Periodic Management and Quality Report

This document is an internal report about the project situation including the compliance with the calendar and the scheduled milestones, the use of the resources, the problems found and an assessment report. The report covers the first year of the project.

7.3 D1.3: 2nd Periodic Management and Quality Report

This document is an internal report about the project situation including the compliance with the calendar and the scheduled milestones, the use of the resources, the problems found and an assessment report. The report covers the second year of the project.
7.4 D1.4: 3rd Periodic Management and Quality Report

This document is an internal report about the project situation including the compliance with the calendar and the scheduled milestones, the use of the resources, the problems found and an assessment report. The report covers the third year of the project.

7.5 D1.5: FastFix Final Report

This internal report summarizes the development of the project and provides an overall final assessment.
8 Continuous: Dissemination and Exploitation

<table>
<thead>
<tr>
<th>Continuous: Dissemination and Exploitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>• D9.1: FastFix Brochure and Public Website</td>
</tr>
<tr>
<td>• D9.2: Dissemination Plan and Reports</td>
</tr>
<tr>
<td>• D9.3: Exploitation Plan and Reports</td>
</tr>
<tr>
<td>• D9.4: Collaboration Plan</td>
</tr>
</tbody>
</table>

Figure 8.1: Deliverables of FastFix Dissemination and Exploitation Activities.

Dissemination activities and exploitation of the FastFix results (illustrated in Figure 8.1) are carried out in WP9. Its objectives are to produce and disseminate promotional material for Fastfix, to effectively disseminate FastFix achievements, to assess the market for FastFix results, to coordinate FastFix with related initiatives, and to effectively plan and organize exploitation of FastFix results.

8.1 D9.1: FastFix Brochure and Public Website

The purpose of this deliverable is to introduce the two initial dissemination tools of the project: the project website and the project brochure.

8.2 D9.2: Dissemination Plan and Reports

This deliverable documents the dissemination and awareness activities performed in the project. It is an iterative deliverable, with an updated version every 6 months (D9.2A, D9.2B, D9.2C, and D9.2D). The scope of this deliverable is to illustrate at length the dissemination approach, methodology, operational plan structure, dissemination means and channels used to launch an extensive dissemination and awareness campaign throughout the project period. All the mechanisms required to identify, steer, execute and report the dissemination activities foreseen under the detailed dissemination plan are defined.

8.3 D9.3: Exploitation Plan and Reports

This document reports on the exploitation activities performed in the FastFix project. It is an iterative deliverable, with an updated version every 6 months (D9.3A, D9.3B, D9.3C, and D9.3D). The scope of this deliverable is to illustrate exploitation approach,
methodology, and operational plan structure to be followed and further extended throughout the project. The mechanisms required to identify, steer and execute the exploitation activities foreseen under the exploitation plan are defined.

8.4 D9.4: Collaboration Plan

This report provides the action plan for the collaboration of FastFix with other projects of the SSAI&E community.