Component Certification
Ph.D. Research Plan

Till G. Bay

13th December 2004

Eidgenössische Technische Hochschule Zürich,
Chair of Software Engineering,
ETH Zentrum, RZJ 22, CH-8092 Zürich, Switzerland,
bay@inf.ethz.ch

Supervisor: Prof. Dr. Bertrand Meyer

Ph.D. starting date: 27 November 2003

Ph.D. expected end date: 27 November 2006

Contents

1 Component Certification

Ease of reuse is one of the most important attributes of our engineering discipline. Component based software engineering is a way of practicing and encouraging reuse. The goal of the thesis is to improve component software development by developing component certification techniques and making them available to everyone. Component certification will improve components that will themselves the be reused more often due to the resulting better quality.

The contributions of this thesis to component certification will be in the field of automated component assessment and certification techniques. I will develop an integrated certification environment, the component server, that will feature all certification automata that are developed during the thesis. Along with the component server I will develop a component model for Eiffel. As not all certifiable attributes of components can be assessed automatically, I will show what certification services should be offered by a third party certification organization.

2 Thesis Overview

Component certification is the process of evaluating software components according to a predefined set of criteria. Normally component certification is performed by a certification authority that applies a certification procedure to asses a component in question. A software component is a piece of software that provides a well defined functionality or a service to the users of that component. A software component can be integrated into other software. One can build software that entirely consists of components. The main benefit of using components is the high degree of reuseability.

The set of criteria that is used in component certification allows the user of the component to judge its quality beforehand. Common judging criteria are crucial during the evaluation phase of the software engineering process, as there is often more than one component that can be used for a single functionality. The goal of my thesis is to evaluate existing criteria sets and refine them to establish a lightweight set that can serve as common judging base across technological frontiers. The criteria set should be enriched with tools that automate the judging of the components.

2.1 Component Model Comparison

Today a big variety of different component models exist. In the scope of this thesis I will study the evolution of a selection of commercial component models and then compare the latest generation of the most commonly used models. This technological comparison will allow me to spot pitfalls when developing component models and it will show where certification has to begin in respect to the different component models. The evolution of the following models will be taken into account:

- COM+
- .net
- EJB
- OSGI

Clearly this is a short list of component models and others will have to be studied as well.

Comparison criteria The comparison criteria taken into account when comparing the component models are mentioned in the following list. A more detailed description of the different criteria would go beyond the scope of this research plan, and will therefore be included in the thesis report.

- ABCDE Classification
- Error handling
- Frameworks supporting Component development
- Forms of reuse, Composeability
- Encapsulation (Separation of interface and implementation)
- Binary compatibility, Language independence, Platform independence
- Version control, Identity
- Interface descriptions, Dependency descriptions, Context descriptions, Meta data

2.2 Eiffel Component Model

Resulting from the comparison of the state of the art in component models described in the previous paragraph I will be able to participate in component model development effort of the Eiffel community. As Eiffel is a cross platform technology that strongly emphasises conceptual hygienics it will be very challenging to develop such a component model for Eiffel. A careful evaluation of component versionning techniques will be a central part of the blueprint for an Eiffel component model.

2.3 Development of Components

A ongoing effort during my thesis will be the development of software components. The component server is built using component technology and contributing component certification units for it will be one part of the component development effort I am undertaking. A very important second part is the cross platform multimedia and game development library ESDL that I am maintaining.

2.4 Component Discovery

Assessing the quality of components is without question a very important issue computer scientists have to address. However the impact of quality assessment is not big enough, if the information about a component's quality is not available to everybody at virtually no cost. Therefore we develop a component search engine, that will allow users to visualize quality information quickly and exhaustingly.

2.5 Certification Techniques

Resulting from the component model comparison, I will be able to develop certification techniques targeted towards the different models. These techniques will be integrated into the component server that is described below. As mentioned before not all certification can be done automatically on the component server. Manual certification steps will be documented for each of the component models.

2.6 Component Server

The main contribution of this thesis is the component server. The component server will provide a high quality software component construction and maintenance facility. It integrates all automated certification modules and showcases the state of the art of integrated high end software construction. The important subsystems of the component server are listed below:

- sourceforge like software project platform focusing on components
- certification modules
- API that allows integration into local build process
- searchable
- usenet and mailing list transparency
- bugtracking
- wiki support
- code annotation support

This list is the result of a one day brainstorming and will be extended as the evaluation of different platforms similar to the component server progresses. The key feature above however is the API that will allow component developers to integrate the component server seamlessly into their build environment. Other requirements that are already known are listed here:

- scalability
- extendibility
- clean visual appearance

The central issue here is the scalability of the component server. The component server will be designed to run on a cluster of machines, that can then be extended. The requirement for a clean visual appearance is also a key point that will contribute to the component server's success in the community.

3 Tentative Thesis Schedule

The tentative scheduling of this thesis envisions the following two milestones:

First internal release of the component server: October 2005

First public release of the component server: March 2006

The expected end of the thesis is the 27 November 2006.